High-Precision Attitude Estimation for Spacecraft

Control of Complex Systems

Project Report

Group 933

Aalborg University
Electronics and IT

Electronics and IT
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
High-Precision Attitude Estimation for Spacecraft

Theme:
Control of Complex Systems

Project Period:
Fall 2022

Project Group:
Group 933

Participant(s):
Danny Dibbern
Rasmus Skjelsager
Siddharth Pathania

Supervisor(s):
Henrik Schiøler

Copies: 1

Page Numbers: 88

Date of Completion:
December 21, 2022

Abstract:

High-precision attitude estimation is vital for many applications in the space industry. This work studies several variations of the Kalman filter with the MEKF, MMEKF and SMEKF being implemented in simulation for attitude estimation, and the EKF, MEKF and UKF being implemented for sensor calibration. The filters are tested and compared in different scenarios, with different sensor combinations and initial estimation errors. The calibration results show that the attitude independent UKF is able estimate the parameters with an accuracy of 99.98%, compared to the 96.84% accuracy of the EKF. The attitude dependent calibration MEKF is able to estimate with an accuracy of 99.06%. The attitude determination results show that the SMEKF performs best in all scenarios and initial conditions, as it converges faster than the MEKF and MMEKF, which both produce the exact same estimates. The SMEKF and MMEKF are similar in computation time, with both having roughly half the computation time of the MEKF.

Contents

Chapter 1

Introduction

In almost all modern day satellite applications, whether it is a low earth orbit satellite imaging the surface of the earth, or a telescope gazing into the depths of cosmos, it is crucial to know and to be able to control the tri-axial orientation (attitude) of the satellite. The satellite's attitude is estimated using a wide variety of sensors such as star trackers, sun trackers, gyroscopes and magnetometers. The data from all these sensors can be combined using sensor fusion algorithms to get the final attitude estimation.

The mathematics of attitude estimation was developed quickly with the advent of space exploration in the 1970s and with the computational power increasing exponentially and the optimization of the algorithms it wasn't long before the first mission namely Magset was launched in 1978 with satellite attitude estimation. [1] With time, the accuracy and reliability of attitude estimation has improved drastically. Nowadays, modern satellite's can estimate the attitude with an accuracy of about 1 arc second. [2] This, for a low earth orbit satellite orbiting at an altitude of around 450 km, pointing it's instrument at the surface of earth gives an error of just 2 m.

The easiest way to represent a spacecraft's attitude is by using Euler's angles, which can be then represented in a rotation matrix. This describes the rotation of the satellite from a global reference frame, to its local reference frame. The global frames generally used are Earth-Centered/Earth-Fixed Frame (ECEF) or a Local-Vertical/Local-Horizontal Frame (LVLH). But the Euler angles can be vulnerable to singularities and gimbal lock, and attitude matrices that do not suffer from this have the downside of having 9 parameters with 6 constraints. Because of this, a quaternion representation of the attitude is generally preferred.

This project focuses on implementing filters for calibrating sensors used to determine the attitude of a satellite, as well as a few variations of the multiplicative Extended Kalman Filter (MEKF) to estimate the attitude from various sensor's data. All the algorithms are evaluated for accuracy and robustness.

Chapter 2

Satellite Modelling

In this chapter the kinematics of the satellite, along with a model of the several sensors will be presented.

2.1 Quaternion Kinematics

As mentioned in the introduction, in this project the quaternion will be used for attitude parameterization due to it's singularity-free property. The quaternion represents the rotation from a global reference frame to the satellite's local body frame of reference.
The satellite's angular velocity is represented by ω which is a vector with 3 components, each representing the angular velocity at an axis.
The attitude of the satellite Δt seconds after time t is given by $q(t + \Delta t)$ and the rotation from $q(t)$ to $q(t + \Delta t)$ is given by the following.

$$\Delta q = \cos\left(\frac{\theta}{2}\right) + u \sin\left(\frac{\theta}{2}\right) \tag{2.1}$$

Where $u = \frac{\omega}{|\omega|}$ is the axis of rotation and $\theta = |\omega|\Delta t$ is the angle of rotation. [3] Here if $q(t) = q$ is the attitude at time t then $q(t + \Delta t) = \Delta q q$ giving us the quaternion derivative as.

$$\dot{q} = \frac{1}{2}\omega q \tag{2.2}$$

It should be noted that to get the product between $\omega \in \mathbb{R}^3$ and $q \in \mathbb{R}^4$, the velocity is expressed as a pure quaternion $\omega = [0 \ \omega_x \ \omega_y \ \omega_z]$ and as such the hamilton product can be employed.[4]

$$\dot{q} = \frac{1}{2}\begin{bmatrix} -\omega_x q_x - \omega_y q_y - \omega_z q_z \\ \omega_x q_w + \omega_z q_y - \omega_y q_z \\ \omega_y q_w - \omega_z q_x + \omega_x q_z \\ \omega_z q_w + \omega_y q_x - \omega_x q_y \end{bmatrix} \tag{2.3}$$

An Omega operator can be defined as following

$$\Omega(\omega) = \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix} \tag{2.4}$$

and thus the final quaternion kinematics can be obtained,

$$\dot{q} = \frac{1}{2}\Omega q \tag{2.5}$$

2.2 Gyroscope

A gyroscope is used for measuring the angular velocity of the spacecraft. Navigation-grade gyroscopes are highly accurate - often more so than a dynamic model of the spacecraft, and is thus often used as a replacement. Several different types of gyroscopes exist, including spinning-mass, optical, or Coriolis-vibratory gyros. The inner workings of these various types are not the focus of this section however. Instead, a model for simulating and calibrating a gyro is presented.

2.2.1 Model

In [5] Farrenkopf provides a gyro model of the form:

$$\boldsymbol{\omega}(t) = \boldsymbol{\omega}^{\text{true}}(t) + \boldsymbol{\beta}^{\text{true}}(t) + \boldsymbol{\eta}_v(t), \quad \boldsymbol{\eta}_v(t) \sim \mathcal{N}(0, \sigma_v^2 \delta(t-\tau) I_3) \tag{2.6a}$$

$$\dot{\boldsymbol{\beta}}^{\text{true}}(t) = \boldsymbol{\eta}_u(t), \quad \boldsymbol{\eta}_u(t) \sim \mathcal{N}(0, \sigma_u^2 \delta(t-\tau) I_3) \tag{2.6b}$$

where $\boldsymbol{\omega}(t)$ is the measured quantity, $\boldsymbol{\omega}^{\text{true}}(t)$ the true angular velocity, $\boldsymbol{\beta}^{\text{true}}(t)$ the bias of the gyro modelled as a random walk process, and $\boldsymbol{\eta}_v(t)$ is a Gaussian white measurement noise process. The random walk process is driven by another Gaussian white noise process $\boldsymbol{\eta}_u(t)$. Each of these processes have their own spectral density given by variance σ_v and σ_u respectively. The noise is assumed to be isotropic, thus the variance is the same for all axes. Lastly, $\delta(\cdot)$ is the Dirac delta function.

The gyro is assumed to be installed in the spacecraft such that its axes align with the spacecraft axes. This may not be a perfectly valid assumption as mechanical inaccuracies for example due to assembly or stresses and vibrations during launch can cause misalignments of the axes. The gyro model in Eq. 2.6 can be extended to include scaling factors and axis misalignments. This extension presumes that imperfections in the gyroscope (mechanical or otherwise) or in the assembly affect

the measurements in a deterministic fashion. Such imperfections can be modelled by:

$$\boldsymbol{\omega}(t) = (I_3 + S)\boldsymbol{\omega}^{\text{true}}(t) + \boldsymbol{\beta}^{\text{true}}(t) + \boldsymbol{\eta}_v(t), \quad \boldsymbol{\eta}_v(t) \sim \mathcal{N}(0, \sigma_v^2 \delta(t-\tau) I_3) \tag{2.7a}$$

$$\dot{\boldsymbol{\beta}}^{\text{true}}(t) = \boldsymbol{\eta}_u(t), \quad \boldsymbol{\eta}_u(t) \sim \mathcal{N}(0, \sigma_u^2 \delta(t-\tau) I_3) \tag{2.7b}$$

where the added term $(I + S)$ in Eq. 2.7a models possible scaling and rotation of the axes in the gyro. The matrix S thus contain the deviations from the perfect model. In general, S is defined as:

$$S = \begin{bmatrix} s_1 & k_{U_1} & k_{U_2} \\ k_{L_1} & s_2 & k_{U_3} \\ k_{L_2} & k_{L_3} & s_3 \end{bmatrix} \tag{2.8}$$

If $S = 0_{3\times 3}$, Eq. 2.7 reduces to the simpler model in Eq. 2.6.

2.2.2 Discretization

The above model is a continuous-time model, but in practice gyros are sampled at discrete intervals. To simulate a gyro on a computer, a discrete-time model is therefore required. Given some sampling interval Δt, the discretization is given by:

$$\boldsymbol{\omega}_{k+1} = \boldsymbol{\omega}_{k+1}^{\text{true}} + \frac{1}{2}\left(\boldsymbol{\beta}_{k+1}^{\text{true}} + \boldsymbol{\beta}_k^{\text{true}}\right) + \left(\frac{\sigma_v^2}{\Delta t} + \frac{1}{12}\sigma_u^2 \Delta t\right)^{\frac{1}{2}} \boldsymbol{N}_{v_k} \tag{2.9a}$$

$$\boldsymbol{\beta}_{k+1}^{\text{true}} = \boldsymbol{\beta}_k^{\text{true}} + \left(\sigma_u^2 \Delta t\right)^{\frac{1}{2}} \boldsymbol{N}_{u_k} \tag{2.9b}$$

for the simple gyro model in Eq. 2.6 or by:

$$\boldsymbol{\omega}_{k+1} = (I_3 + S)\boldsymbol{\omega}_{k+1}^{\text{true}} + \frac{1}{2}\left(\boldsymbol{\beta}_{k+1}^{\text{true}} + \boldsymbol{\beta}_k^{\text{true}}\right) + \left(\frac{\sigma_v^2}{\Delta t} + \frac{1}{12}\sigma_u^2 \Delta t\right)^{\frac{1}{2}} \boldsymbol{N}_{v_k} \tag{2.10a}$$

$$\boldsymbol{\beta}_{k+1}^{\text{true}} = \boldsymbol{\beta}_k^{\text{true}} + \left(\sigma_u^2 \Delta t\right)^{\frac{1}{2}} \boldsymbol{N}_{u_k} \tag{2.10b}$$

for the extended gyro model in Eq. 2.7 [6]. In both these models, the continuous-time Gaussian white noise models $\boldsymbol{\eta}_v(t)$ and $\boldsymbol{\eta}_u(t)$ have been replaced by the discrete-time Gaussian white noise processes $\boldsymbol{N}_{v_k} \sim \mathcal{N}(0, I_3)$ and $\boldsymbol{N}_{u_k} \sim \mathcal{N}(0, I_3)$.

2.3 Magnetometer

A magnetometer is a sensor that measures the strength and direction of a magnetic field. In space, specifically low earth orbit, a magnetometer can be used to determine the orientation of a spacecraft by measuring the direction of the Earth's

magnetic field and comparing this to a reference magnetic field from models such as the International Geomagnetic Reference Field (IGRF).[7] The direction of the magnetic field changes depending on the spacecraft's location on the Earth's surface.

2.3.1 Model

A simple measurement model of the magnetometer is

$$B_k = A(q_k^{\text{true}})R_k + \beta^{\text{true}} + \eta_k, \qquad \eta_k \sim \mathcal{N}(0, \Sigma_k) \tag{2.11}$$

where A_k is the true attitude matrix rotating the measurement into the body frame, R_k is the magnetic field vector in the inertial frame and β^{true}, and η_k are the measurement bias, and Gaussian white noise with covariance Σ_k, respectively. A difference from the gyroscope is that the bias is assumed constant.

Similarly to the gyroscope in section 2.2, it can be assumed that the magnetometer is manufactured with mechanical inaccuracies resulting in misalignment of the axes. Some axes of the magnetometer may also be slightly more sensitive than others which is modelled by the scaling factors. Combined, these can be modelled by:

$$B_k = (I_{3\times 3} + D^{\text{true}})^{-1}(\mathcal{O}^T A(q_k^{\text{true}})R_k + \beta^{\text{true}} + \eta_k) \tag{2.12}$$

where D^{true} is a matrix of scaling factors and non-orthogonality corrections, and $\mathcal{O}$ contains misalignments.

$$D^{\text{true}} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{12} & D_{22} & D_{23} \\ D_{13} & D_{23} & D_{33} \end{bmatrix} \tag{2.13}$$

The matrix D^{true} can be assumed symmetric, since any skew-symmetric contribution can be absorbed into the matrix $\mathcal{O}^T$. [8]
Modelling the magnetometer in this way allows for attitude-independent calibration by estimating D^{true} and β_k. This is described more in depth in section 4.3.

2.4 Star Tracker

A star tracker is a sensor used in satellites to estimate the attitude using the stars in the night sky as reference. It is basically a digital camera with mostly CMOS based sensors used with a fixed focal plane. A modern star tracker can have an accuracy of few arcseconds in the axes perpendicular to the boresight and larger errors for the rotation about the boresight.

2.4.1 Modes of Operation

A star tracker generally has two modes of operation. Initial attitude detection and tracking mode. The initial attitude detection, also known as lost in space mode is used to get the initial attitude without any prior information. The whole Field of View (FOV) is searched for bright star clusters and compared with the entries in a star catalog using sophisticated pattern matching algorithms to estimate the spacecraft's attitude.
In the tracking mode the sensor is tracking a set number of stars. At each time interval the sensor collects light in a region of interest(ROI) where it thinks the tracked stars might be, based on prior attitude estimations and angular velocity data. From the bright spots in the ROI the new position of the tracked stars can be measured and the satellite's attitude can be estimated. If one of the tracked stars move out of the FOV then the tracker searches for another star to track.

2.4.2 Field of View

A star tracker's optics are usually designed with the focal plane centered along its optical axis with a square $N_{pixel} \times N_{pixel}$ resolution. The Field of View of the tracker depends on the focal length of the optics as well as the size of CMOS sensor. A smaller FOV is preferred to get better accuracy in cross-axis attitude measurements. But it is also desirable to track at least 4 stars to get a good estimation. This also dictates the size of the star catalog.
The number of stars in FOV can be modelled as a Poisson distribution, so the probability of N stars in FOV is given by

$$P(N) = e^{-\bar{N}} \frac{\bar{N}^N}{N!} \tag{2.14}$$

Here, $\bar{N}$ is the average number of stars in FOV. So for $\bar{N} = 10$ gives us a 99% chance to have 4 stars in FOV.

2.4.3 Pinhole Model

The optics of the star tracker can be modelled as a pinhole camera as shown in the Fig. 2.1.

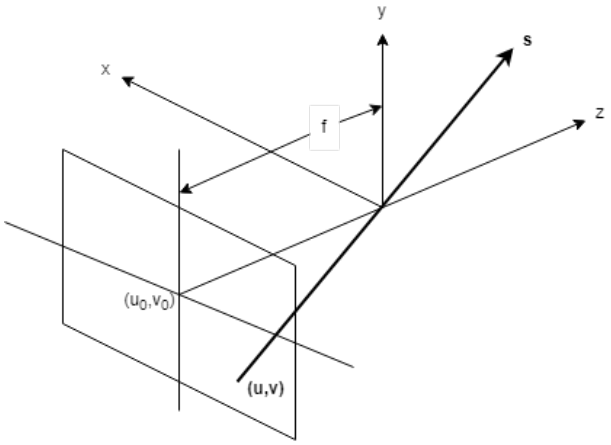

Figure 2.1: Geometry of the star tracker

The local reference frame of the satellite's origin is chosen at the vertex point of the optics with the z axis aligned with the optical axis which is the tracker's boresight. The focal plane is f distance behind the vertex. The focal plane has a (u, v) coordinate system and the center of it where the optical axis meets focal plane is (u_0, v_0). Thus the unit vector **s** from the satellite's local frame to the star can be calculated by Eq. 2.15. [6]

$$s = \frac{1}{\sqrt{f^2 + (u - u_o)^2 + (v - v_o)^2}} \begin{bmatrix} u - u_0 \\ v - v_o \\ f \end{bmatrix} \tag{2.15}$$

2.4.4 Static Attitude Determination

The TRIAD (Tri Axial Attitude Determination) algorithm is the simplest method to calculate the attitude of a satellite. To determine a satellite's attitude at least 3 independent measurements are required. These measurement can be obtained from different sensors such as star tracker, sun sensor and magnetometer. In this case measurements from a star tracker tracking 3 distinct stars in its FOV are used to determine the attitude.
Fig. 2.2 shows unit vector pointing to a star S_l in the local reference frame O_L calculated using Eq. 2.15 and also the unit vector S_g in the global reference frame O_G. The satellite's attitude is represented by the rotational matrix R from the global to the reference frame.

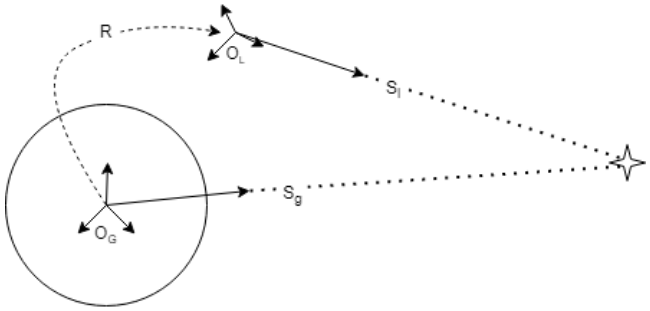

Figure 2.2: Unit vectors to a star in different frames of reference

Eq. 2.18 describes the conversion from one frame of reference to another.

$$S_l = RS_g \tag{2.16}$$

With 3 measurements from the 3 different tracked stars we get.

$$\begin{bmatrix} S_{l1} & S_{l2} & S_{l3} \end{bmatrix} = R \begin{bmatrix} S_{g1} & S_{g2} & S_{g3} \end{bmatrix} \tag{2.17}$$

Where S_{li} and S_{gi} are the unit vector in local and global reference respectively to the *ith* star. From this we can calculate the attitude using the following.

$$R = \begin{bmatrix} S_{l1} & S_{l2} & S_{l3} \end{bmatrix} \begin{bmatrix} S_{g1} & S_{g2} & S_{g3} \end{bmatrix}^{-1} \tag{2.18}$$

But Eq. 2.18 is extremely sensitive to measurement noise and can even violate the orthogonality constraint of the rotation matrix.[6]

Wahba's Problem

The static attitude determination above can be improved upon by using more measurements and by putting different weights on the measurements. This can be done by solving the optimizing problem proposed by Grace Wahba.[9] The problem is to find the orthogonal attitude matrix A with $|A| = 1$, which minimizes the cost function.

$$L(A) = \frac{1}{2} \sum_{i=1}^{N} a_i \left\| b_i - Ar_i \right\| \tag{2.19}$$

Here b_i is a set of satellite local frame measurement and r_i is the corresponding global reference frame vector. a_i are the arbitrary weights given to each measurement. The cost function can be rewritten, by using the orthogonality of A and the unit norm of the vectors as.

$$L(A) = \lambda_o - tr(AB^T) \tag{2.20}$$

where

$$\lambda_o = \sum_{i=1}^{N} a_i \tag{2.21}$$

And B is the attitude profile matrix defined by

$$B = \sum_{i=1}^{N} a_i b_i r_i^T \tag{2.22}$$

As λ_o is a constant, the solution to the optimization problem is the attitude matrix that minimizes the loss function or which maximizes $tr(AB^T)$.

Davenport's q method

Davenport provided with a quaternion solution to the Wahba's problem of attitude determination in [10]. By substituting the attitude matrix with a quaternion representation in 2.20, the loss function can be rewritten as.

$$L(A(q)) = \lambda_o - q^T K(B) q \tag{2.23}$$

Where $K(B)$ is a 4×4 symmetric traceless matrix built with.

$$K(B) = \begin{bmatrix} tr(B) & z^T \\ z & B + B^T - tr(B) I_3 \end{bmatrix} \tag{2.24}$$

Here z is

$$z = \begin{bmatrix} B_{23} - B_{32} \\ B_{31} - B_{13} \\ B_{12} - B_{21} \end{bmatrix} = \sum_{i=1}^{N} a_i (b_i \times r_i) \tag{2.25}$$

As $K(B)$ is a real symmetric matrix, and its eigen value decomposition can be written as.

$$K(B) = \sum_{i=1}^{4} \lambda_i q_i q_i^T \tag{2.26}$$

Here, q_i is the eigenvector corresponding to the eigenvalue λ_i. Also as the trace of $K(B)$ is zero, the sum of the eigen values is also zero.
By substituting 2.26 into 2.23, and expanding it in terms of the maximum eigen value(λ_1) we get.

$$\begin{aligned} L(A(q)) &= \lambda_o - \sum_{i=1}^{4} \lambda_i (q^T q_i)^2 \\ &= \lambda_o - \lambda_1 + \sum_{i=2}^{4} (\lambda_1 - \lambda_i)(q^T q_i)^2 \end{aligned} \tag{2.27}$$

The loss function is minimized if the optimized quaternion($\hat{q}$) is the normalized eigen vector(q_1) of K corresponding to its maximum eigen value(λ_1).

$$\hat{q} = q_1 \tag{2.28}$$

2.5 Fine Sun Sensor

A fine sun sensor, similar to a star tracker is a camera sensor but with a pinhole and a wide field of view of typically $128° \times 128°$. It returns the sun's direction in the local body frame. A simple measurement model would be:

$$S_k = A(q_k^{\text{true}})r_k + \beta_k + \eta_k \tag{2.29}$$

with true attitude matrix A, inertial frame vector measurement r, measurement bias β and measurement noise η.

A reference vector is calculated using models such as the Jet Propulsion Lab's Planetary and Lunar Ephemerides model[11], where the satellite's position with respect to earth along with the current date and time are inputs.[12]

The sun sensor is not available all the time as earth might obstruct the sun from the sensor as seen in Fig. 2.3. In the umbra region, earth is completely covering the sun and no measurement from the sun sensor will be available. On the other hand in penumbra region, the sun is partially visible and can be used for measurements but with a slightly greater error.

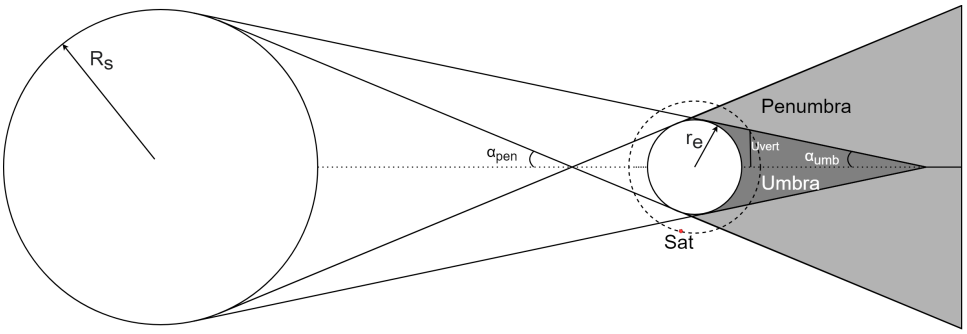

Figure 2.3: Geometery of the star tracker

A low to medium earth orbit satellite will be in the penumbra region for a very short amount of time, as such in this model a binary approach is taken where only if the satellite is in the umbra region, the sensor measurement becomes unavailable. The value for α_{umb} is a constant and can be calculated as follows.[13]

$$\sin(\alpha_{umb}) = \frac{r_s + r_p}{R_p} \tag{2.30}$$

Where R_s is the radius of the sun, r_e is the radius of the planet and D is the distance between the planet and the earth. Whether the satellite is in the umbra region or

not, can be checked by comparing the vertical position of the satellite and the umbra, which is done as shown below:

$$PV = \frac{r_e D}{R_s - r_e} \tag{2.31}$$

$$AB = \sqrt{r^2 - r_p^2} \tag{2.32}$$

Where r is the height of the satellite from the centre of earth.

$$UmbVert = \frac{r_e(PV - AB\cos(\alpha_{umb}))}{PV} \tag{2.33}$$

If Sun_v is the vector from the center of earth to the sun, Sat_v is the vector from earth to the satellite and θ is the angle between them then:

$$SatVert = |Sun_v \times Sat_v| \tag{2.34}$$

```
InUmbra = false
If theta > pi/2 and SatVert < UmbVert
    InUmbra = true
```

4

2.6 Satellite Orbit

A simple satellite circular orbital model is developed which is used in the magnetometer and the fine sun sensor models. The orbit can be described by the satellite's altitude from the center of the earth r and the orbit's angle from the equator $\theta \in [-\pi/2, \pi/2]$ using the following set of equations.

$$x = r\cos(\phi) \tag{2.35}$$

$$y = r\sin(\phi)\cos(\theta) \tag{2.36}$$

$$z = -r\sin(\phi)\sin(\theta) \tag{2.37}$$

Here x, y, and z represent the 3D position of the satellite in earth reference frame. ϕ varies from $[0, 2\pi]$ depending on the satellite's angular velocity ω which can be calculated as:

$$\omega = \frac{v}{r} \tag{2.38}$$

where v is the satellites' translational velocity:

$$v = \sqrt{\frac{GM_e}{r}} \tag{2.39}$$

with G as the gravitational constant and M_e as the mass of the earth.

Then ϕ for a given initial angle ϕ_0 can be propagated by:

$$\phi = \phi_0 + \Delta t \omega \tag{2.40}$$

A plot of an orbit resulting from this model can be seen in Fig. 2.4.

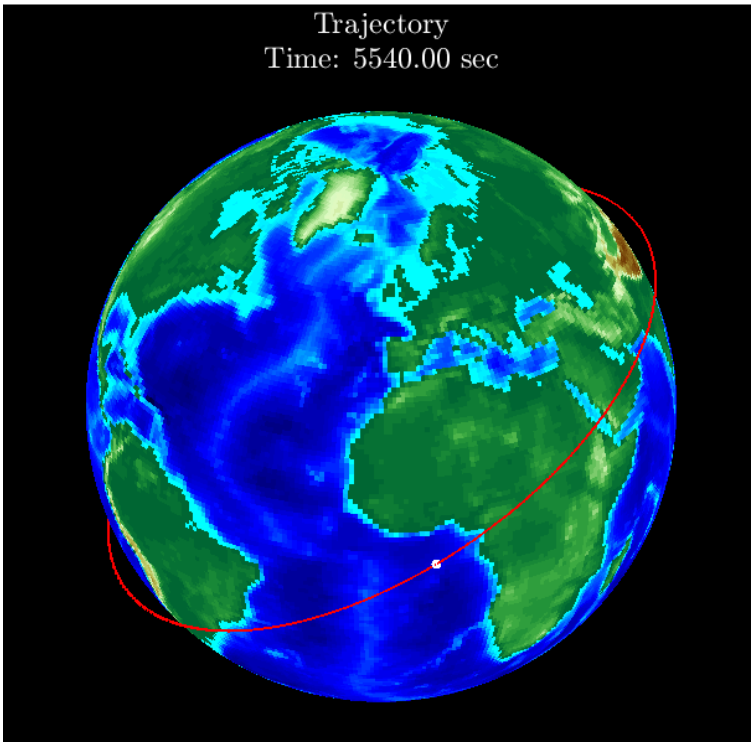

Figure 2.4: Plot of a simulated circular orbit after a single full orbit. The red curve displays the trajectory of the satellite depicted by the white dot.

2.7 Geomagnetic Field Model

To simulate the geomagnetic field for making magnetometer measurements, the 13th generation of the International Geomagnetic Reference Field [7] is used. The field is a magnetic dipole with some deviations (anomalies), depending on location, with the largest anomalies occuring over Brazil and Siberia [14]. The field can be expressed as:

$$V(r,\theta,\phi,t) = a \sum_{n=1}^{k} \left(\frac{a}{r}\right)^{n+1} \sum_{m=0}^{n} \left(g_n^m(t)\cos(m\phi) + h_n^m(t)\sin(m\phi)\right) P_n^m(\theta), \tag{2.41}$$

where $a = 6371.2$ km is the reference radius of the earth, r is the radial distance from the center of the earth, θ is the geocentric latitude with ϕ the longitude. The function $P_n^m(\theta)$ is a Legendre function of degree n and order m (see [7] for more details). The coefficients $g_n^m(t)$ and $h_n^m(t)$ are known as the Gauss coefficients. The coefficients are time dependent and are given in the IGRF. A few magnetic field lines are plotted in Fig. 2.5.

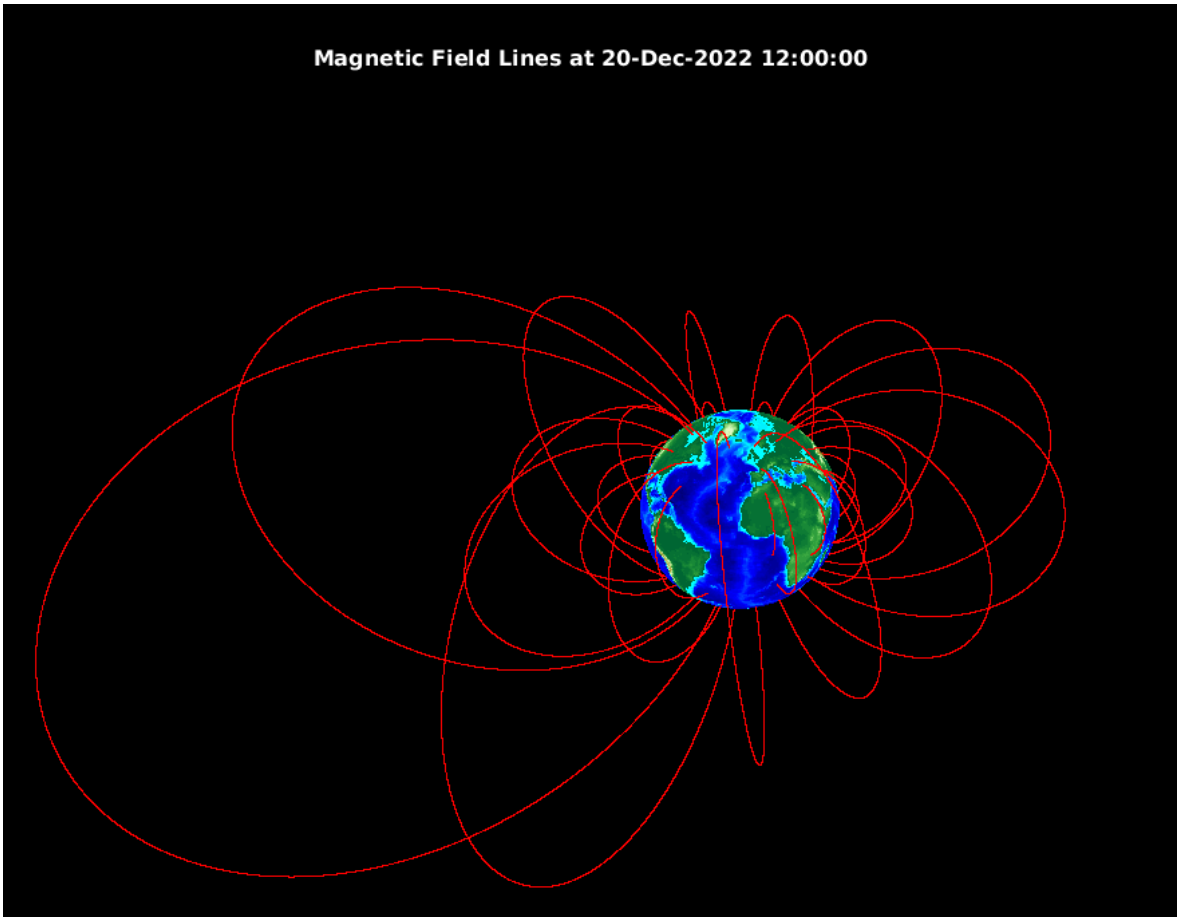

Figure 2.5: Plot of geomagnetic field llines simulated with the IGRF-13 model.

Chapter 3

Attitude Estimation

In many modern applications of satellites, an ultra-high estimate of the spacecraft attitude is neccesary, to be able to control and point at targets that are at great distances. To this end, the spacecraft is fitted with several sensors such as star trackers, gyroscopes, magnetometers etc. But as the sensor measurements are noisy, they can not be relied on directly for the attitude estimation. Instead it is neccesary to process the different measurements in a filter, that estimates and corrects for the noise.
In this chapter, different variations of the Kalman filter will be described, with the ones found most relevant for spacecraft attitude estimation being implemented and tested in the following chapters.

3.1 Kalman Filter

As the filter will be implemented in an embedded system, a microcontroller running on the satellite, it will be working within the discrete time domain. Consider the discrete linear system of difference equations:

$$x_{k+1} = Ax_k + Bu_k + w_k, \qquad w_k \sim \mathcal{N}(0,\, Q_k) \tag{3.1a}$$

$$y_k = Cx_k + v_k, \qquad v_k \sim \mathcal{N}(0,\, R_k) \tag{3.1b}$$

with white noise processes w_k and v_k, and covariance matrices Q_k and R_k. Where A, B and C are implicitly discretized. The covariance matrices are usually unknown, but an initial prediction is needed for the update and propagation steps. They can then be treated as tuning parameters of the filter.[15]

Initial

The Kalman filter is initialized with an initial estimate of the covariance matrix for measurements and estimated states

$$P(t_0) = P_0 = P_{k=1}^{-} \tag{3.2}$$

along with a vector of the initial estimated states

$$\hat{x}(t_0) = \hat{x}_0 = \hat{x}_{k=1}^{-} \tag{3.3}$$

Gain

First the Kalman gain is calculated as

$$K_k = P_k^- C^T \left(C P_k^- C^T + R_k \right)^{-1} \tag{3.4}$$

Update

With the new measurements, the estimate of the states are then updated by an addition of the previous estimate with a product of the Kalman gain and measurement residual

$$\hat{x}_k^+ = \hat{x}_k^- + K_k \left(y_k - C\hat{x}_k^- \right) \tag{3.5}$$

The estimate of the covariance matrix is then updated using the Kalman gain

$$P_k^+ = \left(I - K_k C \right) P_k^- \left(I - K_k C \right)^T + K_k R_k K_k \tag{3.6}$$

Propagation

After the measurements have been used to update the gain, state- and covariance estimates, the estimated states have to be propagated to next timestep

$$\hat{x}_{k+1}^- = A\hat{x}_k^+ + Bu_k \tag{3.7}$$

Similarly the covariance estimate is propagated by

$$P_{k+1}^- = A P_k^+ A^T + Q_k \tag{3.8}$$

3.2 Extended Kalman Filter

To be able to accurately estimate the states of a highly non-linear system, it is usually not adequate to only use a single linearization. As the states move further away from the operating point, the linearized model will become increasingly inaccurate, this is especially clear when the model includes sines/cosines. The extended Kalman filter deals with this by linearizing with a dynamic operating point equal to the current system states, at each iteration of the filter.[16]

Consider the discrete time non-linear system of difference equations:

$$x_{k+1} = f(x_k, u_k) + w_k, \qquad w_k \sim \mathcal{N}(0,\, Q_k) \tag{3.9a}$$

$$y_k = h(x_k) + v_k, \qquad v_k \sim \mathcal{N}(0,\, R_k) \tag{3.9b}$$

Linearization

At each iteration of the filter, a local linearization is performed. The state-transition matrix, F_k, is derived through the Jacobian, evaluated at the current estimate of the system states $\hat{x}_k$ and the current input u_k.

$$F_k = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\hat{x}_k^-,\, u_k} \tag{3.10}$$

The output-sensitivity matrix is derived similarly as

$$H_k = \left.\frac{\partial h(x)}{\partial x}\right|_{\hat{x}_k^-} \tag{3.11}$$

Gain

The Kalman gain is updated at each iteration by

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{3.12}$$

using the initial or propagated covariance matrix P_k^- along with the updated output-sensitivity matrix H_k and the measurement covariance matrix R_k.

Update

The state estimate is then updated by addition of the propagated estimate with a measurement residual weighted by the updated Kalman gain

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(y_k - h(\hat{x}_k^-)) \tag{3.13}$$

The estimated covariance matrix is updated lastly, using the updated output-sensitivity matrix H_k

$$P_k^+ = P_k^- - P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} H_k P_k^- \tag{3.14}$$

Propagation

Finally the updated state estimates and covariance matrix are be propagated to the next time step. Unlike the linear Kalman filter, the EKF utilizes the non-linear model for propagation of the states

$$\hat{x}_{k+1}^- = f(x_k^+, u_k) \tag{3.15}$$

while the covariance matrix is propagated similarly to the linear Kalman filter

$$P_{k+1}^- = F_k P_k^+ F_k^T + Q_k \tag{3.16}$$

3.2.1 Representational Considerations

Three-Component Representations

The most intuitive way to represent rotations are three-component, as only 3 parameters are needed to represent the attitude in three-dimensional space. The issue with these representations are that they all have discontinuities and singularities, making them unsuitable for estimation in a Kalman filter unless it can be guaranteed that the filter avoids these singular points. Quaternions avoids the singularities by using 1 real and 3 imaginary components to represent rotations. This is the lowest dimensionality, singularity-free rotation representation, but it comes with it's own issues which will addressed in the following sections.

Quaternion Representation

For quaternions to be used as a representation for rotations, it is a requirement that they preserve their unity norm. In the additive quaternion representation, the quaternion update step is:

$$\hat{q}_k^+ = \hat{q}_k^- + K_k[y_k - h_k(\hat{x}_k^-)] \tag{3.17}$$

It is clear that this step would cause issues with preserving the unity norm, as there would have to be a special relation between $\hat{q}_k^-$ and $K_k[y - h(\hat{x}_k^-)]$, which is not the case. One solution to this would be brute normalization. [6]

3.3 Multiplicative EKF

The main idea of the multiplicative EKF (MEKF) is to not estimate the quaternion directly, but instead estimate a local 3-component attitude error $\delta\vartheta_{3\times1}$ along with non-attitude parameters $\delta\xi_{m\times1}$. This three-component representation avoids discontinuities by updating a global quaternion estimate, and resetting the local attitude error estimate, thus keeping the three-component attitude error small and far away from singular points. The product of the attitude error quaternion and the quaternion estimate

$$q_{true} = \delta q(\delta\vartheta) \otimes \hat{q} \tag{3.18}$$

is the true quaternion. As seen later, this is also used to update the global attitude estimate instead of the additive update in Eq. 3.17, hence the name multiplicative EKF. The clear advantage of the MEKF over the additive EKF is the dimensionality reduction from $4 + m$ to $3 + m$, which results in a more intuitive covariance matrix along with obvious computational advantages, thus the AEKF will not be considered further.

Initial

The MEKF is initialized with a covariance matrix, similar to the KF and EKF

$$P(t_0) = P_0 = P_{k=1}^- \tag{3.19}$$

and a local state vector consisting of estimated errors

$$\Delta x(t_0) = \Delta x_0 = \begin{bmatrix} \delta\vartheta_{k=1}^- \\ \delta\xi_{k=1}^- \end{bmatrix} = \begin{bmatrix} 0_{3\times 1} \\ 0_{n-3\times 1} \end{bmatrix} \tag{3.20}$$

where $\delta\vartheta$ is the attitude errors and $\delta\xi$ the non-attitude errors. A global quaternion estimate is also initialized

$$\hat{q}(t_0) = \hat{q}_0 = \hat{q}_{k=1}^- \tag{3.21}$$

but is not directly estimated in the MEKF, but instead updated outside the filter.

Gain

At each iteration of the MEKF, the output-sensitivity matrix H_k is updated using the newest measurements.

$$H_k = \begin{bmatrix} A(\hat{q}_k^-)r_1 & 0_{3\times n-3} \\ \vdots & \vdots \\ A(\hat{q}_k^-)r_N & 0_{3\times n-3} \end{bmatrix} \tag{3.22}$$

where N is the number of measurements, and r is the measured vector in the inertial frame. In the same manner as the EKF, the updated output-sensitivity matrix and propagated covariance matrix is then used to calculate the current Kalman gain

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{3.23}$$

Update

After updating the Kalman gain, it is used to update the covariance matrix by subtracting the weighted propagated covariance matrix $K_k H_k P_k^-$ from the propagated covariance matrix P_k^-

$$P_k^+ = [I_{n\times n} - K_k H_k] P_k^- \tag{3.24}$$

The state vector Δx is updated as in the EKF and KF, by addition of a weighted vector measurement residual with the propagated state vector.

$$\Delta x_k^+ = \begin{bmatrix} \delta\vartheta_k^+ \\ \delta\xi_k^+ \end{bmatrix} = \begin{bmatrix} \delta\vartheta_k^- \\ \delta\xi_k^- \end{bmatrix} + K_k[y_k - h_k(\hat{q}_k^-)] \tag{3.25}$$

where

$$y_k = \begin{bmatrix} A(q_k^{\text{true}})r_1 \\ \vdots \\ A(q_k^{\text{true}})r_N \end{bmatrix} + \eta_k \tag{3.26a}$$

and

$$h_k(q_k^-) = \begin{bmatrix} A(\hat{q}_k^-)r_1 \\ \vdots \\ A(\hat{q}_k^-)r_N \end{bmatrix} \tag{3.26b}$$

or with direct quaternion measurement and Rodrigues vector parameterization the state update becomes

$$\Delta x_k^+ = \begin{bmatrix} \delta\vartheta_k^- \\ \delta\xi_k^- \end{bmatrix} + K_k[2\frac{(q_k^m \otimes \hat{q}_k^{-1})_{1:3}}{(q_k^m \otimes \hat{q}_k^{-1})_4} - 2\frac{(\hat{q}_k^- \otimes \hat{q}_k^{-1})_{1:3}}{(\hat{q}_k^- \otimes \hat{q}_k^{-1})_4} + \begin{bmatrix} I_{3\times 3} & 0_{3\times n} \end{bmatrix} \Delta x_k^-] \tag{3.27}$$

where $\delta\xi_k$ is the non-attitude parameters to be estimated, q_k^m is the quaternion measurement and $\hat{q}_k^{-1}$ is understood to be the inverse of $\hat{q}_k^-$. The global quaternion estimate can then be updated by a product of the updated attitude error quaternion and the previous quaternion estimate.

$$\hat{q}_k^+ = \delta q(\delta\vartheta_k^+) \otimes \hat{q}_k^- \tag{3.28}$$

where the attitude error quaternion is constructed as

$$\delta q(\delta\vartheta_k^+) = \begin{bmatrix} \frac{1}{2}\delta\vartheta_k^+ \\ 1 \end{bmatrix} \tag{3.29}$$

finally the non-attitude parameters are updated

$$\xi_k^+ = \xi_k^- + \delta\xi_k^+ \tag{3.30}$$

Propagation

An estimate of the angular velocity $\hat{\omega}$ is needed for the quaternion kinematic propagation

$$\hat{\omega}_k = \omega_k^m - \xi_k^+ \tag{3.31}$$

where ω_k^m is the gyroscope measurement, and ξ_k could be the gyroscope bias and/or scaling factors and misalignments.

After correcting the gyroscope measurement, the estimated angular velocity is used to propagate the estimated quaternion to the next time step:

$$\hat{q}_{k+1}^- = \Theta_k \hat{q}_k^+ \tag{3.32}$$

with the discrete quaternion propagation matrix Θ

$$\Theta_k = \begin{bmatrix} \cos\left(\frac{1}{2}\|\hat{\omega}_k\|\Delta t\right) I_{3\times 3} - [\phi_k\times] & \phi_k \\ -\phi_k^T & \cos(\frac{1}{2}\|\hat{\omega}_k\|\Delta t) \end{bmatrix} \tag{3.33}$$

where

$$\varphi_k = \sin\left(\frac{1}{2}\|\hat{\omega}_k\|\Delta t\right)\frac{\hat{\omega}_k}{\|\hat{\omega}_k\|} \tag{3.34}$$

The estimated covariance matrix is then propagated to the next timestep:

$$P_{k+1}^- = \Phi_k P_k^+ \Phi_k^T + Q_k \tag{3.35}$$

with the discrete state transition matrix Φ and discrete covariance matrix Q_k. If the given model is continuous, with state transition matrix $F(t)$ and process noise spectral density Q, the discrete matrices can be discretized with:

$$\Phi_k = I_{n\times n} + \Delta t F(t) \tag{3.36}$$

where the state transition matrix $F(t)$ can take different forms depending on the amount and type of sensors used in the filter.

$$Q_k = \Delta t G_k Q G_k^T \tag{3.37}$$

similarly the process noise distribution matrix $G(t)$ also looks different depending on the sensor setup. The specific matrices for the mission mode and calibration filters will be shown later in chapter 4.

Reset

As part of the propagation, the errors states are reset to zero to avoid having to propagate $\delta\vartheta$ and $\delta\xi$. This is done by first moving the information contained in the local error states into the global states $\hat{q}$ and ξ as in Eqs. 3.28 and 3.29. And then resetting the local error states by: [6]

$$\Delta x_{k+1}^- = \left[0_{n\times 1}\right] \tag{3.38}$$

3.4 Murrel's MEKF

The purpose of Murrel's formulation of the MEKF, is to make it more suitable for real time onboard computation. This is done by dealing with expensive computations such as large matrix inversion.

In the computation of the MEKF gain in Eq. 3.23, a $3N \times 3N$ matrix inversion has to be performed, where N is the amount of vector measurements.

$$[H_k P_k^- H_k^T + R_k]^-_{3N\times 3N} \tag{3.39}$$

with

$$H_k = \begin{bmatrix} A(\hat{q}_k^-)r_1 & 0_{3\times n-3} \\ \vdots & \vdots \\ A(\hat{q}_k^-)r_N & 0_{3\times n-3} \end{bmatrix}_{3N\times n} \tag{3.40}$$

As seen in Eq. 3.25, despite the EKF involving non-linear models a linear update is still performed. The principle of superposition can then be used to update the covariance matrix in Eq. 3.24, and the states in Eq. 3.25 by instead using one 3×1 vector measurement to compute the gain, and performing the update step N times.

$$[H_k P_k^- H_k^T + R_k]^-_{3\times 3} \tag{3.41}$$

with

$$H_k = \begin{bmatrix} A(\hat{q}_k^-)r_i & 0_{3\times n-3} \end{bmatrix}_{3\times n} \tag{3.42}$$

After the N updates, the reset is performed to move the updated values into the global estimate. An overview of the MMEKF is shown in Tab. 3.1.

Table 3.1: Discrete Murrel's Multiplicative Extended Kalman Filter

Initialize	$P(t_0) = P_{k=1}^-,\ \hat{q}(t_0) = \hat{q}_{k=1}^-$
	$\Delta x(t_0) = \Delta x_{k=1}^- = 0_{n\times 1}$
Gain	**for** $i = 1 : N$
	$H_{k,i} = \begin{bmatrix} A(\hat{q}_k^-)r_i & 0_{3\times n-3} \end{bmatrix}$
	$K_{k,i} = P_{k,i}^- H_{k,i}^T (H_{k,i} P_{k,i}^- H_{k,i}^T + R_k)^{-1}$
Update	$P_{k,i}^+ = [I_{n\times n} - K_{k,i} H_{k,i}] P_{k,i}^-$
	$\Delta x_{k,i}^+ = \Delta x_{k,i}^- + K_{k,i}(y_{k,i} - h_{k,i}(q_k^-))$
	if $i < N$
	$P_{k,i+1}^- = P_{k,i}^+$
	$\Delta x_{k,i+1}^- = \Delta x_{k,i}^+$
	end
	end
	$\hat{q}_k^+ = \delta q(\delta\vartheta_{k,N}^+) \otimes \hat{q}_k^-$
	$\xi_k^+ = \xi_k^- + \delta\xi_k^+$
Propagation	$\hat{\omega}_k = \omega_k^m - \xi_k^+$
	$\hat{q}_{k+1}^- = \Theta_k(\hat{\omega}_k)\hat{q}_k^+$
	$P_{k+1}^- = \Phi_k P_{k,N}^+ \Phi_k^T + Q_k$
Reset	$\Delta x_{k+1}^- = \begin{bmatrix} 0_{n\times 1} \end{bmatrix}$

This variation of the MEKF changes the inversion of a $3N \times 3N$ matrix, to an inversion of a 3×3 matrix N times. The computation time in Murell's MEKF is thus significantly reduced, due to the computation time increasing exponentially with the size of the matrix. [6]

3.5 Sequential MEKF

The sequential MEKF is a modification of Murrel's MEKF, where the point is to improve the estimation accuracy, by updating the quaternion estimate with each measurement and resetting the error state vector after each measurement update. Such that the quaternion estimate $\hat{q}^-$ in Eq. 3.42 is propagated from $i-1$ and not $k-1$, which results in using a more accurate quaternion estimate for the linearization of the output sensitivity matrix, thus improving the filter accuracy.[17] An overview of the SMEKF can be seen in Tab. 3.2.

Table 3.2: Discrete Sequential Multiplicative Extended Kalman Filter

Initialize	$P(t_0) = P^-_{k=1},\ \hat{q}(t_0) = \hat{q}^-_{k=1}$
	$\Delta x(t_0) = \Delta x^-_{k=1} = 0_{n\times 1}$
Gain	**for** $i = 1 : N$
	$H_{k,i} = \begin{bmatrix} A(\hat{q}^-_{k,i})r_i & 0_{3\times n-3} \end{bmatrix}$
	$K_{k,i} = P^-_{k,i}H^T_{k,i}(H_{k,i}P^-_{k,i}H^T_{k,i} + R_k)^{-1}$
Update	$\Delta x^+_{k,i} = \Delta x^-_{k,i} + K_{k,i}(y_{k,i} - h_{k,i}(\hat{q}^-_{k,i}))$
	$\hat{q}^+_{k,i} = \delta q(\delta\vartheta^+_{k,i}) \otimes \hat{q}^-_{k,i}$
	$\hat{\xi}^+_{k,i} = \hat{\xi}^-_{k,i} + \delta\xi^+_{k,i}$
	if $i < N$
	$\hat{q}^-_{k,i+1} = \hat{q}^+_{k,i}$
	$\hat{\xi}^-_{k,i+1} = \hat{\xi}^+_{k,i}$
Reset	$\Delta x^-_{k,i+1} = \begin{bmatrix} 0_{n\times 1} \end{bmatrix}$
	end
	end
	$P^+_k = [I_{n\times n} - K_{k,N}H_{k,N}]P^-_k$
Propagation	$\hat{\omega}_k = \omega^m_k - \hat{\xi}^+_{k,N}$
	$\hat{q}^-_{k+1} = \Theta_k(\hat{\omega}_k)\hat{q}^+_k$
	$P^-_{k+1} = \Phi_k P^+_k \Phi^T_k + Q_k$

3.6 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an alternative to the EKF [16][18]. Unlike the EKF it does not rely on locally linearizing the nonlinear state transition and measurement functions, f and h. Rather, it generates a small set of statistically significant points (called sigma points) and propagates those points through the nonlinear functions. The propagated points are then used to compute a weighted mean as the state estimate, with a corresponding weighted error covariance matrix. This process is called the Unscented Transform (UT).

3.6.1 The Unscented Transform

Several different types of unscented transforms exist. They differ in how many sigma points are generated and how the weights are computed. Only one type will be presented here, the Scaled UT as presented in [16] and [18]. The first step is to generate the sigma points. With $x \in \mathbb{R}^n$, a total of $2n+1$ sigma points will be generated. Given a mean $\hat{x}$ and an error covariance P_{xx}, the sigma points are selected such that

$$\hat{x} = \sum_{i=0}^{2n} W_i \chi_i \tag{3.43}$$

$$P_{xx} = \sum_{i=0}^{2n} W_i (\chi_i - \hat{x})(\chi_i - \hat{x})^T \tag{3.44}$$

This leads to selecting the sigma points as:

$$\chi_0 = \hat{x} \tag{3.45a}$$

$$\chi_i = \hat{x} + \sqrt{n+\lambda} c_i, \qquad i = 1, \dots, n \tag{3.45b}$$

$$\chi_{n+i} = \hat{x} - \sqrt{n+\lambda} c_i, \qquad i = 1, \dots, n \tag{3.45c}$$

where c_i is the ith column of the Cholesky factor C_{xx} of the error covariance matrix $P_{xx} = C_{xx} C_{xx}^T$, and λ is a tunable parameter defined as

$$\lambda = \alpha^2 (n + \kappa) - n \tag{3.46}$$

which can be tuned by choice of α and κ. Furthermore, the weights are given by:

$$W_0^{[\hat{y}]} = \frac{\lambda}{n+\lambda} \tag{3.47a}$$

$$W_0^{[P_{yy}]} = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta) \tag{3.47b}$$

$$W_i = \frac{1}{2(n+\kappa)}, \qquad i = 1, \dots, 2n \tag{3.47c}$$

where $W_0^{[\hat{y}]}$ is used for computing the mean, and $W_0^{[P_{yy}]}$ for computing the error covariance. This also introduces a third tuning parameter, β which may be chosen arbitrarily.
The sigma points χ_i are transformed by the nonlinear function and the resulting transformed points are defined:

$$\xi_i = g(\chi_i) \tag{3.48}$$

The resulting mean and error covariance of the unscented transform is approximated with:

$$\hat{y} = W_0^{[\hat{y}]}\xi_0 + \sum_{i=1}^{2n} W_i\xi_i \tag{3.49}$$

$$P_{yy} = W_0^{[P_{yy}]}(\xi_0 - \hat{y})(\xi_0 - \hat{y})^T + \sum_{i=1}^{2n} W_i(\xi_i - \hat{y})(\xi_i - \hat{y})^T \tag{3.50}$$

where $\hat{y}$ is the estimate of the mean of the transformed distribution and P_{yy} the estimate of the error covariance.

3.6.2 Filter equations

The UT procedure can be used to obtain state and measurement estimates in a Kalman filter by applying the state transition function $f(x_k, u_k)$ and the measurement function $h(x_k)$ as the nonlinear function $g(\cdot)$. Consider the following discrete-time system model:

$$x_{k+1} = f(x_k, u_k) + w_k, \qquad w_k \sim \mathcal{N}(0,\, Q_k) \tag{3.51a}$$

$$z_k = h(x_k) + v_k, \qquad v_k \sim \mathcal{N}(0,\, R_k) \tag{3.51b}$$

The UT is used to obtain the estimates $\hat{x}_{k+1}^-$ and $\hat{z}_k$ [16]. The Kalman gain matrix is usually computed as

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{3.52}$$

The matrix P_k^- is computed from the output P_{xx} of the UT when obtaining $\hat{x}_k^-$ with f and $\hat{x}_{k-1}^+$ by:

$$P_k^- = P_{xx} + Q_k \tag{3.53}$$

and the product $H_k P_k^- H_k^T$ is obtained from the UT of $\hat{x}_k^-$ with the nonlinear function h, where $\hat{z}_k$ is estimated:

$$P_{zz} = H_k P_k^- H_k^T. \tag{3.54}$$

Lastly, the cross-covariance term $P_k^- H_k^T$ is approximated by:

$$P_k^- H_k^T \approx P_{xz} = \sum_{i=0}^{2n} W_i(\chi_i - \hat{x}_k^-)(\xi_i - \hat{z}_k)^T \tag{3.55}$$

which leads to the Kalman gain being computed as:

$$K_k = P_{xz}(P_{zz} + R_k)^{-1}. \tag{3.56}$$

The update equations for the UKF are then given by

$$\hat{x}_k^+ = \hat{x}_k^- + K_k(z_k - \hat{z}_k) \tag{3.57a}$$
$$P_k^+ = P_k^- - K_k P_{xz}^T \tag{3.57b}$$

3.6.3 Square-root formulation

In order to improve numerical stability a square-root formulation of the UKF has been developed [18]. The square-root formulation differs from the regular UKF by updating and propagating the Cholesky factor of the error covariance matrix rather than the error covariance matrix itself. This method for computing the Cholesky factors relies on QR factorization and a procedure for updating the Cholesky factor, described in [19]. The "thin" QR factorization of a matrix $A \in \mathbb{R}^{m\times n}$ is given by

$$A^T = Q_1 R_1 \tag{3.58}$$

where $Q_1 \in \mathbb{R}^{m\times n}$ has orthonormal columns and R_1 is upper triangular, and $R_1 = C^T$ where C is the lower triangular Cholesky factor of $P = AA^T$. Then

$$R_1^T R_1 = AA^T = P \tag{3.59}$$

If given C, the lower triangular Cholesky factor of symmetric positive definite $P = CC^T$, then efficient methods for updating the Cholesky factorization when P is updated by

$$\tilde{P} = P + wvv^T, \quad v \in \mathbb{R}^n, \quad w \in \mathbb{R} \tag{3.60}$$

with z a vector, w some scalar weight. In the UT, the error covariance matrix of the transformed distribution (see Eq. 3.50) is defined as a sum of scaled outer products, similar to $\sqrt{w}vv^T$. To obtain the lower Cholesky factor of P_{yy} it is possible to take the QR factorization

$$Q_1 R_1 = Q_1 C^T = \sqrt{W_1}\begin{bmatrix}\xi_1 - \hat{y} & \dots & \xi_{2n} - \hat{y} & \sqrt{\mathcal{R}}\end{bmatrix} \tag{3.61}$$

where $\sqrt{\mathcal{R}}$ is the matrix square root of the noise covariance matrix (typically $\sqrt{Q_k}$ during propagation and $\sqrt{R_k}$ at the measurement update). To include the remaining sigma point in the computation of the factorization, the update is applied:

$$\tilde{P} = P^T + \sqrt{W_0^{[P_y y]}}(\xi_0 - \hat{y})(\xi_0 - \hat{y})^T \tag{3.62}$$

and the updated Cholesky factor $\tilde{C}$ can be found as described in [19]. This operation can be denoted simply

$$\tilde{C}_k = \text{cholupdate}\left(C_k, (\chi_0 - \hat{x}), \sqrt{W_0^{[P_y y]}}\right). \tag{3.63}$$

The above applies to how the unscented transform is computed. However, the computation of the Kalman gain and the subsequent post-measurement update of the error covariance also changes. In the standard formulation the Kalman gain was given by Eq. 3.56. The square-root formulation of the UT computes the Cholesky factor of $P_{zz} + R_k$. The Kalman gain is therefore given by

$$K_k = P_{xz}(C_{z_k} C_{z_k}^T)^{-1}. \tag{3.64}$$

Since C_{z_k} is triangular, this can be solved efficiently with forward-backward substitution. The posterior error covariance of the state estimate in the standard formulation is given by Eq. 3.57b. In the square-root formulation, a matrix U is formed

$$U = K_k C_{z_k} \tag{3.65}$$

and with each column u_i in U as the update/downdate vector a series of succesive Cholesky downdates is applied to C_k^- to obtain C_k^+.

$$C_k^+ = \text{cholupdate}\left(C_k^-, U, -1\right). \tag{3.66}$$

where the weight -1 is used because the downdate is defined as:

$$\tilde{P} = P - vv^T \tag{3.67}$$

Applying these methods result in a more numerically stable algorithm, and if implemented correctly may also be faster [18].

Chapter 4

Implementation

In this chapter the implementation of the simulation of various sensors and filters will be described.

4.1 Star Tracker

The star tracker simulation takes the true quaternion attitude of the satellite as input to generate the snap shot of the stars that the tracker would have taken. In the snapshot, noise can be added to simulate the sensor's optical aberrations among other sources of noise. From this snapshot the attitude can be calculated back and the propagation of the noise to the final measurement can be observed.

4.1.1 Star Catalog

The star catalog used in the simulation is a set of uniformly distributed randomly generated points on a sphere. All the points are simulated to be a constant 25 light years away from the satellite. The number of stars in the catalog is chosen to be 3000 which gives an average number of stars in an FOV of 15° to be 13 and the probability of finding at least 4 stars in the FOV to be greater than 99%. [6]

4.1.2 Generating the SnapShot

The z-axis of the local reference frame is assumed as the optical axis and is calculated with respect to the global reference frame using the inverse rotation of the true quaternion.

$$O_x = A((q_{true})^{-1}) \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}' \tag{4.1}$$

The stars that are in the FOV of the tracker are found by calculating the angle between star's unit vector (S_g) and the optical axis (O_x).

$$\theta = \cos^{-1}(S_g \cdot O_x) \tag{4.2}$$

All the stars in the FOV are then projected onto a plane using central projection of a sphere as shown in fig 4.1.

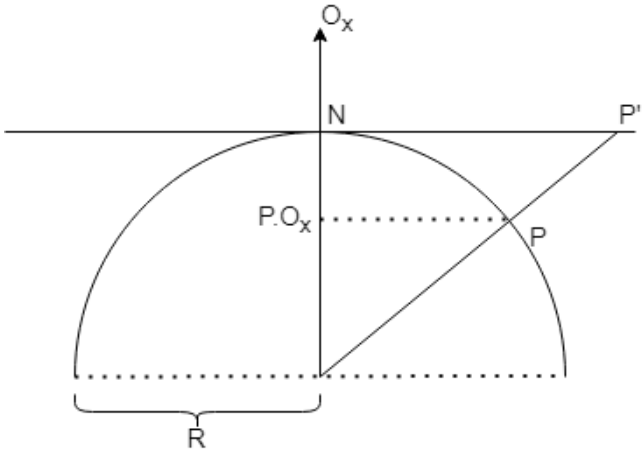

Figure 4.1: Central Projection of a sphere on a plane

This projection can be calculated by using the following eq.

$$P' = P\frac{R}{P \cdot O_x} \tag{4.3}$$

This 2D projection is still off center and can be centered by subtracting it with N which is the point where the optical axis meets the star sphere.

$$P'' = P' - N \tag{4.4}$$

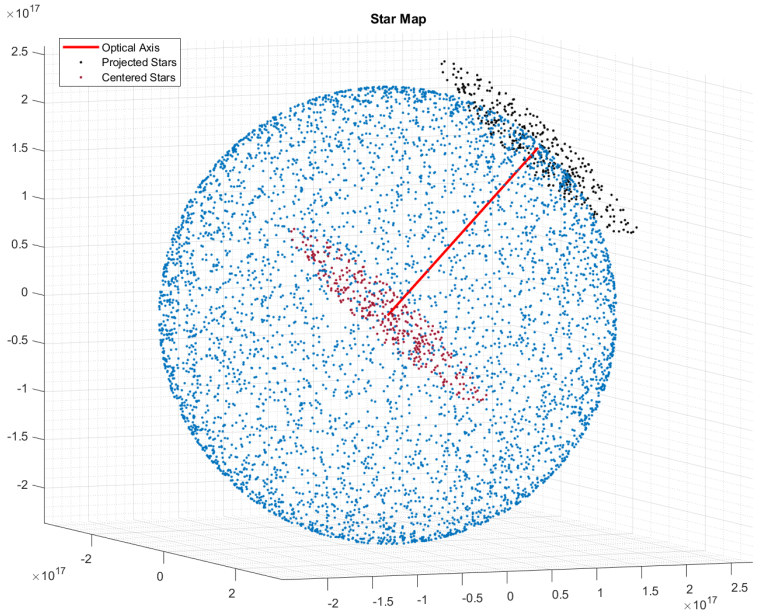

Figure 4.2: Projected Stars in the FOV of 60°

Fig 4.2 shows the stars in the FOV projected on a plane and then moved to the center. But these centered points are still in 3D and needs to be rotated into the xy-plane.This is done by rotating the plane such that the optical axis aligns with the z-axis.
For this rotation, axial angle representation is used. The axis of rotation is the cross product between the z-axis and the optical axis. And the angle of the rotation is the angle between z-axis and the optical axis. The same can be seen in the following code snippet.

```
%Rotate the projecte plane into xy axis(MAKE IT 2D)
u = cross(optic_axis, [0 0 1]);
ang = atan2(norm(cross(optic_axis,[0 0 1])),dot(optic_axis,[0 0 1]));
rotquat = quaternion(axang2quat([u (ang)]));
StarsIn2D = rotatepoint(rotquat,CenteredStars);
```

Fig. 4.3 shows the 2D snapshot obtained after the rotation but this still does not take into account the boresight rotation and the pinhole camera model.

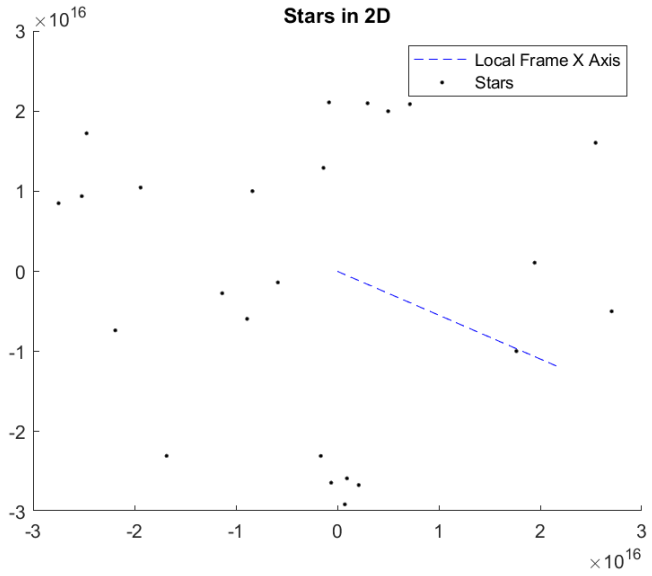

Figure 4.3: Stars In 2D

The correct boresight rotation is calculated from the local frame x-axis which is projected on the global xy-plane as shown in Fig. 4.3. For the 2D rotation, the rotation matrix is used as shown below.

$$Rot_m = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{4.5}$$

$$P_{rot} = Rot_m P'' \tag{4.6}$$

Here θ is the angle between the body frame x-axis projected on the 2D plane and the global x-axis.
Due to the pinhole model now the snapshot is to be cropped, inverted and scaled to fit the square camera sensor. The scaling factor is calculated as follows.

$$SensorSize = \frac{resolution \times 2.54}{ppi \times 100} \tag{4.7}$$

Here $resolution$ is the resolution of the sensor which generally is 512×512 or 1024×1024. The ppi is the pixel density of the sensor in pixels per inch.
For a given FOV and sensor size the focal length f will be fixed and can be calculated as follows.

$$f = \frac{SensorSize}{2 \tan\left(\frac{Fov}{2}\right)} \tag{4.8}$$

Now the scaling factor can be calculate as the ratio of the focal length and the radius of the star catalog sphere i.e. 25 light years.
Therefore we can obtain the final snapshot as seen in fig 4.4.

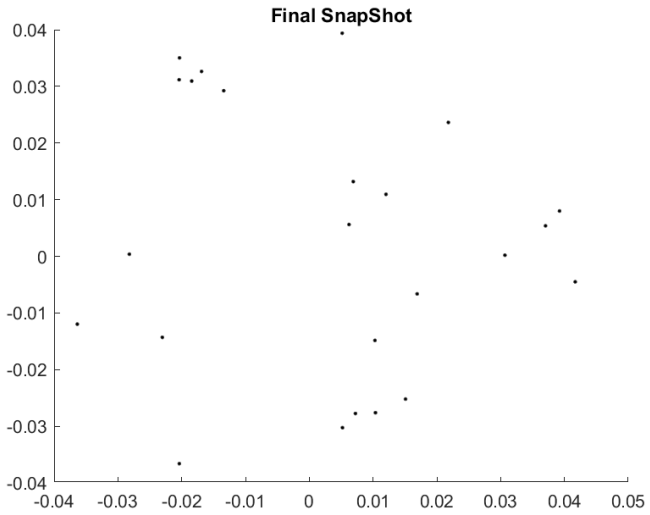

Figure 4.4: Final Snap Shot

Now noise can be artificially added based on the accuracy of the Star tracker which is defined by the parameter K_{cent} in pixels. The final error is randomly generated in the snapshot as shown below.

```
1 err = SensorSize*Kcent/resolution;
2 snap = snapshot_rot * -photo_scale + randn(1,2) * Err;
```

4.1.3 Determining Attitude

The satellite attitude is determined using Davenport's q method as described in sec 2.4.4. The attitude profile matrix(B) is calculate from the set of local measurements obtained from the snapshot and the corresponding reference vectors from the star catalog. From B, the symmetric traceless matrix K can be calculated according to the eq 2.24. The satellite attitude estimated is the eigen vector corresponding to the maximum eigen value of K.

```
% Determining Attitude
B = zeros(3,3);
for i = 1:n
    B = B + GlobalToStar(i,:)'*LocalToStar(i,:);
end
Z = [B(2,3)-B(3,2);B(3,1)-B(1,3);B(1,2)-B(2,1)];
K = [trace(B),Z';Z,B+B'-trace(B)*eye(3)];
[q_meas,~] = eigs(K,1);
```

4.2 Gyroscope Calibration MEKF

In order to get as accurate results as possible it is necessary to calibrate the gyroscope. This section describes an implementation of the MEKF that calibrates the gyro, as presented in [6]. The calibration is performed mainly to determine the scaling and misalignment factors in the gyro model of Eq. 2.7. That is, the goal is to estimate the matrix S. We call this estimate $\hat{S}$ and define the matrices as:

$$\hat{S} = \begin{bmatrix} \hat{s}_1 & \hat{k}_{U_1} & \hat{k}_{U_2} \\ \hat{k}_{L_1} & \hat{s}_2 & \hat{k}_{U_3} \\ \hat{k}_{L_2} & \hat{k}_{L_3} & \hat{s}_3 \end{bmatrix} \tag{4.9}$$

Arranging these variables into vectors we have:

$$\hat{\boldsymbol{s}} = \begin{bmatrix} \hat{s}_1 & \hat{s}_2 & \hat{s}_3 \end{bmatrix}^T \tag{4.10a}$$

$$\hat{\boldsymbol{k}}_U = \begin{bmatrix} \hat{k}_{U_1} & \hat{k}_{U_2} & \hat{k}_{U_3} \end{bmatrix}^T \tag{4.10b}$$

$$\hat{\boldsymbol{k}}_L = \begin{bmatrix} \hat{k}_{L_1} & \hat{k}_{L_2} & \hat{k}_{L_3} \end{bmatrix}^T \tag{4.10c}$$

Corresponding vectors can be defined for the true matrix S. Typically, the entries in the matrix S are modelled as stochastic processes with dynamics defined by:

$$\dot{\boldsymbol{s}} = \boldsymbol{\eta}_s \tag{4.11a}$$

$$\dot{\boldsymbol{k}}_U = \boldsymbol{\eta}_U \tag{4.11b}$$

$$\dot{\boldsymbol{k}}_L = \boldsymbol{\eta}_L \tag{4.11c}$$

where $\boldsymbol{\eta}_s$, $\boldsymbol{\eta}_U$, and $\boldsymbol{\eta}_L$ are random vectors with spectral densities given by $\sigma_s^2 I_3$, $\sigma_U^2 I_3$, and $\sigma_L^2 I_3$ respectively. Under the assumption that the scaling as misalignment factors are constants, these random vectors can be assumed to have zero mean, and zero variance, thus becoming deterministic 0 vectors. This will be applied in the implementation, however, for completeness they will be described without such assumptions in the following.

The filter will estimate quantities $\hat{\boldsymbol{s}}$, $\hat{\boldsymbol{k}}_U$, and, $\hat{\boldsymbol{k}}_L$ along with the attitude $\hat{q}$ and the gyro bias $\hat{\boldsymbol{\beta}}$. As previously described, the MEKF first estimates the error state, that is, the change Δx in the states x. Thus, we write the error state as:

$$\Delta x(t) = \begin{bmatrix} \delta\boldsymbol{\vartheta}(t) \\ \Delta\hat{\boldsymbol{\beta}}(t) \\ \Delta\boldsymbol{s}(t) \\ \Delta\boldsymbol{k}_U(t) \\ \Delta\boldsymbol{k}_L(t) \end{bmatrix}_{n\times 1} \tag{4.12}$$

The system model for the continuous-time error state with discrete-time measurements is given by:

$$\Delta\dot{x}(t) = f\left(t, x(t)\right) + G(t)w(t) \tag{4.13a}$$

$$y_k = h(x_k) + v_k \tag{4.13b}$$

where the input vector $w(t)$ has spectral density $Q(t)$ and is given by:

$$w(t) = \begin{bmatrix} \boldsymbol{\eta}_v^T(t) & \boldsymbol{\eta}_u^T(t) & \boldsymbol{\eta}_s^T(t) & \boldsymbol{\eta}_U^T(t) & \boldsymbol{\eta}_L^T(t) \end{bmatrix}^T \tag{4.14}$$

and the measurement noise has covariance matrix R_k. This covariance matrix depends on the particular sensor. With this system model, the matrices $F(t)$, $G(t)$, $Q(t)$, and H_k required for the MEKF must be developed.
Starting by the matrix $F(t)$, it is given by:

$$F(t) = \begin{bmatrix} -[\hat{\boldsymbol{\omega}}(t)\times] & -(I_{3\times 3} - \hat{S}) & -\mathrm{diag}(\boldsymbol{\omega}(t) - \hat{\boldsymbol{\beta}}) & -\hat{U} & -\hat{L} \\ 0_{n-3\times 3} & 0_{n-3\times 3} & 0_{n-3\times 3} & 0_{n-3\times 3} & 0_{n-3\times 3} \end{bmatrix} \tag{4.15}$$

The first block is simply the propagation of the change in attitude which is given by the cross-product matrix of the angular velocity when ignoring terms of higher than first order [6]. The second to fifth blocks in $F(t)$ arise from the gyro model. The measurement from the gyro is given in Eq. 2.7a. Isolating the true angular velocity one obtains the expression:

$$\boldsymbol{\omega}^{\text{true}}(t) = (I_3 + S)^{-1}\left(\boldsymbol{\omega}(t) - \boldsymbol{\beta}^{\text{true}}(t) - \boldsymbol{\eta}_v(t)\right) \tag{4.16}$$

which for small S is well approximated by:

$$\boldsymbol{\omega}^{\text{true}}(t) = (I_3 - S)\left(\boldsymbol{\omega}(t) - \boldsymbol{\beta}^{\text{true}}(t) - \boldsymbol{\eta}_v(t)\right). \tag{4.17}$$

The estimate of the true angular velocity is therefore given by:

$$\hat{\boldsymbol{\omega}} = (I_3 - \hat{S})(\boldsymbol{\omega} - \hat{\boldsymbol{\beta}}) \tag{4.18}$$

Defining $\boldsymbol{\delta\omega} = \boldsymbol{\omega}^{\text{true}} - \hat{\boldsymbol{\omega}}$ and ignoring second-order terms, the following is obtained:

$$\boldsymbol{\delta\omega} = -\text{diag}(\boldsymbol{\omega} - \hat{\boldsymbol{\beta}})\Delta\boldsymbol{s} - \hat{U}\Delta\boldsymbol{k}_U - \hat{L}\Delta\boldsymbol{k}_L - (I_3 - \hat{S})(\Delta\boldsymbol{\beta} + \boldsymbol{\eta}_v) \tag{4.19}$$

where

$$\hat{U} = \begin{bmatrix} \omega_2 - \hat{\beta}_2 & \omega_3 - \hat{\beta}_3 & 0 \\ 0 & 0 & \omega_3 - \hat{\beta}_3 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.20}$$

$$\hat{L} = \begin{bmatrix} 0 & 0 & 0 \\ \omega_1 - \hat{\beta}_1 & 0 & 0 \\ 0 & \omega_1 - \hat{\beta}_1 & \omega_2 - \hat{\beta}_2 \end{bmatrix} \tag{4.21}$$

in which the subscripted numbers denote the corresponding element of the vector. From Eq. 4.19, the blocks in F_k can be found by inspection. In general, the matrix $G(t)$ is given by the block diagonal matrix:

$$G(t) = \text{blkdiag}\Big(\begin{bmatrix} -(I_3 - \hat{S}) & I_3 & I_3 & I_3 & I_3 \end{bmatrix} \Big) \tag{4.22}$$

However, under the assumption that the scaling factors and misalignments are constant, it can be reduced to:

$$G(t) = \begin{bmatrix} -(I_{3\times3} - \hat{S}) & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} \\ 0_{n-6\times3} & 0_{n-6\times3} \end{bmatrix} \tag{4.23}$$

since there is no process noise for the constants in this case. The spectral density of the process noise is given by:

$$Q(t) = \text{blkdiag}\Big(\begin{bmatrix} \sigma_v^2 I_3 & \sigma_u^2 I_3 & \sigma_s^2 I_3 & \sigma_U^2 I_3 & \sigma_L^2 I_3 \end{bmatrix} \Big) \tag{4.24}$$

which can be reduced similarly to $G(t)$. Lastly, given that the measurements are vector measurements, the measurement model $h(x_k)$ is given by:

$$h(x_k) = \begin{bmatrix} A(\hat{q}_k^-)r_1 \\ \vdots \\ A(\hat{q}_k^-)r_N \end{bmatrix} \tag{4.25}$$

and linearization results in the matrix H_k, given by:

$$H_k = \begin{bmatrix} [A(\hat{q}_k^-)r_1\times] & 0_{3\times n-3} \\ \vdots & \vdots \\ [A(\hat{q}_k^-)r_N\times] & 0_{3\times n-3} \end{bmatrix} \tag{4.26}$$

In order to execute the filter on an onboard computer, the model must be discretized. This is achieved by discretizing the matrix $F(t)$ and the product $G(t)Q(t)G^T(t)$ with a simple forward-Euler approximation:

$$\Phi_k \approx I_n + \Delta t F(t) \tag{4.27a}$$
$$Q_k \approx \Delta t G(t)Q(t)G^T(t) \tag{4.27b}$$

This discretization provides sufficiently accurate results for use in a discrete Extended Kalman Filter.

4.2.1 Smoothing

If it is possible to store batches of estimates, state transition matrices, and error covariance matrices, smoothing can be applied to the filter outputs to improve the results if necessary [6]. The Rauch-Tung-Striebel optimal Kalman smoother can applied for this purpose [20]. It smooths the N estimates of the Kalman filter with a backwards process, starting from the last estimate with:

$$\hat{x}_{s_N} = \hat{x}_N^+ \tag{4.28a}$$
$$P_{s_N} = P_N^+ \tag{4.28b}$$

Then, at each step back, a new gain is computed:

$$\mathcal{K}_k = P_k^+ \Phi_k^T (P_{k+1}^-)^{-1} \tag{4.29}$$

which is used to compute the smoothed values:

$$\hat{x}_{s_k} = \hat{x}_k^+ + \mathcal{K}_k(\hat{x}_{s_{k+1}} - \hat{x}_{k+1}^-) \tag{4.30a}$$
$$P_{s_k} = P_k^+ - \mathcal{K}_k(P_{k+1}^- - P_{s_{k+1}})\mathcal{K}_k^T \tag{4.30b}$$

This generally provides more accurate results throughout the time series, with the downside that smoothing cannot be executed in real-time.

4.3 Magnetometer Calibration Filters

The magnetometer model presented in section 2.3 can be used for attitude-independent calibration, meaning the magnetometer can be calibrated without having an estimate of the attitude. This allows the magnetometer to be used for initial attitude determination without information from any other attitude sensors. There are several possible ways to calibrate the magnetometer, using either batch estimation methods such as the TWOSTEP-algorithm of [8], or real-time calibration filters as in [21]. Several real-time calibration filters are presented in [21] including an Extended Kalman Filter (EKF) and an Unscented Kalman Filter (UKF). This section

will consider only real-time filters and both an EKF and a UKF are implemented and their performance will be compared in later chapters.
The purpose of attitude-independent calibration is estimating the quantities D^{true} and β^{true} of Eq. 2.12. The matrix $\mathcal{O}$ is not observable, since rotations that may be attributed to $\mathcal{O}$ are indistinguishable from of the attitude matrix A_k. Thus, the misalignments are not estimated with the following approach.

4.3.1 System model with synthetic measurement

In order to determine the estimates $\hat{D}$ and $\hat{\beta}$, a new, synthetic measurement z_k is introduced and a measurement model for the Kalman filters are derived from this [8] [21]. Let B_k be the measurement from the three-axis magnetometer at time k and R_k be the corresponding value of the measured magnetic field in the world frame coordinates. The value of R_k can be determined with the IGRF model. Then z_k is defined as:

$$z_k = \|B_k\|^2 - \|R_k\|^2 = B_k^T B_k - R_k^T R_k \tag{4.31a}$$
$$= -B_k^T \left[2D^{\text{true}} + \left(D^{\text{true}}\right)^2\right] B_k + 2B_k^T (I_3 + D^{\text{true}})\beta^{\text{true}} - \|\beta^{\text{true}}\|^2 + v_k \tag{4.31b}$$

where v_k is a noise term given by:

$$v_k = 2\left[\left(I_3 + D^{\text{true}}\right) B_k - \beta^{\text{true}}\right]^T \eta_k - \|\eta_k\|^2 \tag{4.32}$$

which is approximately Gaussian with mean and variance given by:

$$\mu_k = -\text{tr}\left(\Sigma_k\right) \tag{4.33a}$$
$$\sigma_k^2 = 4\left[\left(I_3 + D^{\text{true}}\right) B_k - \beta^{\text{true}}\right]^T \Sigma_k \left[\left(I_3 + D^{\text{true}}\right) B_k - \beta^{\text{true}}\right] + 2\text{tr}\left(\Sigma_k^2\right) \tag{4.33b}$$

where Σ_k is the covariance matrix of the magnetometer noise term of Eq. 2.11. We may introduce new variables $\mathbf{S}_k$, E^{true}, and $\mathbf{E}^{\text{true}}$, parametrized by B_k and D^{true}:

$$\mathbf{S}_k = \begin{bmatrix} B_{1_k}^2 & B_{2_k}^2 & B_{3_k}^2 & 2\left(B_{1_k}B_{2_k}\right) & 2\left(B_{1_k}B_{3_k}\right) & 2\left(B_{2_k}B_{3_k}\right)\end{bmatrix}^T \tag{4.34a}$$

$$E^{\text{true}} = 2D^{\text{true}} + \left(D^{\text{true}}\right)^2 = \begin{bmatrix} E_{11}^{\text{true}} & E_{12}^{\text{true}} & E_{13}^{\text{true}} \\ E_{12}^{\text{true}} & E_{22}^{\text{true}} & E_{23}^{\text{true}} \\ E_{13}^{\text{true}} & E_{23}^{\text{true}} & E_{33}^{\text{true}} \end{bmatrix} \tag{4.34b}$$

$$\mathbf{E}^{\text{true}} = \begin{bmatrix} E_{11}^{\text{true}} & E_{22}^{\text{true}} & E_{33}^{\text{true}} & E_{12}^{\text{true}} & E_{13}^{\text{true}} & E_{23}^{\text{true}}\end{bmatrix}^T \tag{4.34c}$$

Now, the state vector x for the Kalman filters is defined:

$$x = \begin{bmatrix} \beta_1^{\text{true}} & \beta_2^{\text{true}} & \beta_3^{\text{true}} & D_{11}^{\text{true}} & D_{22}^{\text{true}} & D_{33}^{\text{true}} & D_{12}^{\text{true}} & D_{13}^{\text{true}} & D_{23}^{\text{true}}\end{bmatrix}^T \tag{4.35}$$

With these definitions Eq. 4.31b is rewritten and the measurement model $h_k(x_k)$ is given by:

$$z_k = h_k(x_k) + v_k = -\mathbf{S}_k^T \mathbf{E}^{\text{true}} + 2B_k^T \left(I_3 + D^{\text{true}}\right) \beta^{\text{true}} - \|\beta^{\text{true}}\|^2 + v_k \tag{4.36}$$

By assuming the parameters constant the process noise $w = 0$ and the state transition function is given by:

$$x_{k+1} = f_k(x_k) = x_k \tag{4.37}$$

These equations make up the model used by the filters to compute $\hat{D}$ and $\hat{\beta}$.

4.3.2 Extended Kalman Filter

In addition to the presented state and measurement functions the EKF requires that the Jacobians of the functions be derived. The Jacobian of $f(x)$ is trivial, since:

$$F_k(x) = \frac{\partial f}{\partial x} = I \tag{4.38}$$

The Jacobian of $h(x)$ is given by:

$$H_k(x) = \frac{\partial h}{\partial x} = \begin{bmatrix} 2B_k^T(I_3 + D) - 2\beta^T & -S^T M_{ED}(\hat{D}) + 2J_k(\hat{x}) \end{bmatrix} \tag{4.39}$$

where the parameters $J_k(\hat{x})$ and $M_{ED}(\hat{D})$ are given by

$$J_k(\hat{x}) = \begin{bmatrix} B_{1_k}\hat{\beta}_1 & B_{2_k}\hat{\beta}_2 & B_{3_k}\hat{\beta}_3 & B_{1_k}\hat{\beta}_2 + B_{2_k}\hat{\beta}_1 & B_{1_k}\hat{\beta}_3 + B_{3_k}\hat{\beta}_1 & B_{2_k}\hat{\beta}_3 + B_{3_k}\hat{\beta}_2 \end{bmatrix} \tag{4.40}$$

$$M_{ED}(\hat{D}) = 2I_6 + \begin{bmatrix} 2\hat{D}_{11} & 0 & 0 & 2\hat{D}_{12} & 2\hat{D}_{13} & 0 \\ 0 & 2\hat{D}_{22} & 0 & 2\hat{D}_{12} & 0 & 2\hat{D}_{23} \\ 0 & 0 & 2\hat{D}_{33} & 0 & 2\hat{D}_{13} & 2\hat{D}_{23} \\ \hat{D}_{12} & \hat{D}_{12} & 0 & \hat{D}_{11} + \hat{D}_{22} & \hat{D}_{23} & \hat{D}_{13} \\ \hat{D}_{13} & 0 & \hat{D}_{13} & \hat{D}_{23} & \hat{D}_{11} + \hat{D}_{33} & \hat{D}_{12} \\ 0 & \hat{D}_{23} & \hat{D}_{23} & \hat{D}_{13} & \hat{D}_{12} & \hat{D}_{22} + \hat{D}_{33} \end{bmatrix} \tag{4.41}$$

Due to the state dynamics of Eq. 4.37, the filter's propagation has no effect. The filter equations are therefore given by:

$$\hat{x}_{k+1} = \hat{x}_k + K_k \left(z_{k+1} - h_{k+1}\left(\hat{x}_k\right)\right) \tag{4.42a}$$

$$P_{k+1} = \left(I_9 - K_k H_{k+1}(\hat{x}_k)\right) P_k \left(I_9 - K_k H_{k+1}(\hat{x}_k)\right)^T + K_k \sigma_{k+1}^2(\hat{x}_k) K_k^T \tag{4.42b}$$

$$K_k = P_k H_{k+1}^T(\hat{x}_k) \left(H_{k+1}(\hat{x}_k) P_k H_{k+1}^T(\hat{x}_k) + \sigma_{k+1}^2(\hat{x}_k)\right)^{-1} \tag{4.42c}$$

Here, the functions $h_{k+1}(\hat{x}_k)$, $H_{k+1}(\hat{x}_k)$, and $\sigma_{k+1}^2(\hat{x}_k)$ are the evaluations of the respective function using B_{k+1} and $\hat{x}_k$.

4.3.3 Unscented Kalman Filter

The UKF implemented for magnetometer calibration uses the same model as the EKF, and is otherwise implemented as described in section 3.6, using the square-root formulation. This was done due to issues with numerical stability of the standard UKF when directly computing Cholesky factorizations.
The tuning parameters are chosen to be equivalent to the ones found in [21] with:

$$\begin{aligned}\alpha &= 0.1\\ \kappa &= 3 - n = -6\\ \beta &= 2\end{aligned}$$

Tuning κ and β has little effect on the results, whereas decreasing α (e.g. to $\alpha = 0.01$ or smaller) is noted to result in very slightly more accurate estimations but at the price of slower convergence.

4.4 Mission Mode MEKF

The filter described in this section is a modification of a filter described by Markley and Crassidis in [6]. This filter works under the assumption that the various sensors are calibrated and that estimating only the attitude and the gyro bias, neither of which are constant, is necessary. It is assumed here that a quaternion measurement of the satellite's attitude is available. Further, it is assumed that the gyroscope measurements are used to propagate the state estimate, by estimating the true angular velocities using the bias estimate (and calibrated values from the calibration filter).

The filter has state vector $\hat{x}$:

$$\hat{x} = \begin{bmatrix}\hat{q}\\ \hat{\beta}\end{bmatrix} \tag{4.43}$$

where $\hat{q}$ is the estimate of the attitude quaternion, and $\hat{\beta}$ is the estimate of the gyro bias. In accordance with section 3.3 the filter estimates the error state $\Delta\hat{x}$, defined as:

$$\Delta x = \begin{bmatrix}\delta\hat{q}(\boldsymbol{a}_g)\\ \Delta\hat{\beta}\end{bmatrix} \tag{4.44}$$

where the true rotation is represented by $q = \delta q(\boldsymbol{a}_g) \otimes q_{\text{ref}}$. Thus, $\delta\hat{q}(\boldsymbol{a}_g)$ is the estimate of the rotation error between $\hat{q}^{(-)}$ (the prior) and $\hat{q}^{(+)}$ (the posterior), parametrized by the three-component Gibbs vector $\boldsymbol{a}_g$:

$$\boldsymbol{a}_g = \frac{2}{q_4}\begin{bmatrix}q_1\\ q_2\\ q_3\end{bmatrix} \tag{4.45}$$

In continuous time, the state propagation matrix is given by:

$$F(t) = \begin{bmatrix} -[\hat{\omega}(t)\times] & -I_{3\times3} \\ 0_{3\times3} & 0_{3\times3} \end{bmatrix} \tag{4.46}$$

This model assumes that the calibrated values of the gyro scaling factor matrix $\hat{S} = S^{\text{true}}$ and that the gyro measurements are already corrected with this, such that the model of the gyro effectively becomes that of Eq. 2.6. Thus, only $-I_{3\times3}$ is necessary for propagating the state. Similarly, for the matrix $G(t)$, which is then given by:

$$G(t) = \begin{bmatrix} -I_{3\times3} & 0_{3\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} \tag{4.47}$$

Lastly, the measurements are assumed to be vector measurements, and thus the matrix H_k is given by:

$$H_k = \begin{bmatrix} [A(\hat{q}_k^-)r_1\times] & 0_{3\times n-3} \\ \vdots & \vdots \\ [A(\hat{q}_k^-)r_N\times] & 0_{3\times n-3} \end{bmatrix} \tag{4.48}$$

In the implementation the filter, the propagation of state and covariance are discretized. Discretization of H_k is not necessary as the measurement model is already discrete. Discretization of state propagation requires discretization of the matrix F. The discretized state propagation matrix Φ has a closed form solution (shown in [6]) which is given by:

$$\Phi = \begin{bmatrix} \Phi_{11} & \Phi_{12} \\ \Phi_{21} & \Phi_{22} \end{bmatrix} \tag{4.49}$$

where

$$\Phi_{11} = I_3 - [\hat{\omega}\times]\frac{\sin(\|\hat{\omega}\|\Delta t)}{\|\hat{\omega}\|} + [\hat{\omega}\times]^2\frac{(1-\cos(\|\hat{\omega}\|\Delta t))}{\|\hat{\omega}\|^2} \tag{4.50a}$$

$$\Phi_{12} = [\hat{\omega}\times]\frac{(1-\cos(\|\hat{\omega}\|\Delta t))}{\|\hat{\omega}\|^2} - I_3\Delta t - [\hat{\omega}\times]^2\frac{(\|\hat{\omega}\|\Delta t - \sin(\|\hat{\omega}\|\Delta t))}{\|\hat{\omega}\|^3} \tag{4.50b}$$

$$\Phi_{21} = 0_{3\times3} \tag{4.50c}$$

$$\Phi_{22} = I_3 \tag{4.50d}$$

In continuous-time, the matrix $Q(t)$ is the spectral density of the process noise. The relation between the continuous-time spectral density and discrete-time covariance matrix is given by:

$$Q_k = \int_0^{\Delta t} \Phi G(t)Q(t)G(t)^T\Phi^T \, \mathrm{d}t = \begin{bmatrix} Q_{11k} & Q_{12k} \\ Q_{12k}^T & Q_{22k} \end{bmatrix} \tag{4.51}$$

A closed form solution of the submatrices of Q_k is then:

$$Q_{11k} = \left(\sigma_v^2 \Delta t\right) I_3 + \sigma_u^2 \left(\frac{1}{3}\Delta t^3 I_3 - [\hat{\omega}\times]^2 \frac{2\|\hat{\omega}\|\Delta t - 2\sin\left(\|\hat{\omega}\|\Delta t\right) - \frac{1}{3}\|\hat{\omega}\|^3 \Delta t^3}{\|\hat{\omega}\|^5} \right) \tag{4.52a}$$

$$Q_{12k} = \sigma_u^2 \left([\hat{\omega}\times] \frac{\left(\|\hat{\omega}\|\Delta t - \sin\left(\|\hat{\omega}\|\Delta t\right)\right)}{\|\hat{\omega}\|^3} - \frac{1}{2}\Delta t^2 I_3 - [\hat{\omega}\times]^2 \frac{\left(\frac{1}{2}\|\hat{\omega}\|^2 \Delta t^2 + \cos\left(\|\hat{\omega}\|\Delta t\right) - 1\right)}{\|\hat{\omega}\|^4} \right) \tag{4.52b}$$

$$Q_{22k} = \left(\sigma_u^2 \Delta t\right) I_3 \tag{4.52c}$$

However, if the sampling rate is below the Nyquist limit, then the submatrices may be approximated:

$$Q_{11k} \approx \left(\sigma_v^2 \Delta t + \frac{1}{3}\sigma_u^2 \Delta t^3\right) I_3 \tag{4.53a}$$

$$Q_{12k} \approx \left(-\frac{1}{2}\sigma_u^2 \Delta t^2\right) I_3 \tag{4.53b}$$

$$Q_{22k} = \left(\sigma_u^2 \Delta t\right) I_3 \tag{4.53c}$$

The Nyquist limit can be determined by the inequality $\|\hat{\omega}\|\Delta t < \frac{\pi}{\xi}$, where ξ is some safety factor (typically $\xi = 10$). Even if the sampling rate is not below the Nyquist limit, the approximation often provides results that are sufficiently accurate while saving computation time. [6]

Chapter 5

Estimation Experiments

In order to evaluate the performance of the different filters, they will have to be compared and tested in different scenarios, with a varying suite of sensors available and under differing initial conditions. The following attributes are of particular interest for the experiments:

- Mean squared error of Euler angle estimates
- Mean squared error of bias estimates
- Computation time

The majority of the experiments will be conducted for the mission mode filters, and are split into the 5 scenarios described further below. The calibration filters will be evaluated based on their deviation from the true bias and scaling factors, and the estimated values will be used for the mission mode experiments.

5.0.1 Mission Mode Experiments

1st Scenario

In the first scenario, the satellite is equipped with only the highest precision sensors available: star trackers and gyroscopes.

2nd Scenario

In the second scenario, we want to evaluate the performance of the filter in a situation where the star tracker measurements are not available, that is the satellite is equipped with: fine sun sensors, magnetometers and gyroscopes.

3rd Scenario

In the third scenario, we want to evaluate the performance of a filter using only the cheapest available sensors: gyroscopes and magnetometers.

4th Scenario

In the fourth scenario, we consider a situation in which measurements will not always be available due to being in an eclipse using only: fine sun sensors and gyroscopes.

5th Scenario

In the fifth scenario, we consider a full suite of sensors being available. This includes all modelled sensors from chapter 2: star trackers, gyroscopes, magnetometers and fine sun sensors.

Initialization

All of the above scenarios will be tested under 4 different initial conditions, to evaluate their robustness to initial attitude estimation errors, with the values as shown in Tab. 5.1:

Initial Attitude Estimation Errors	
Test 1	$\begin{bmatrix}1° & 1° & 1°\end{bmatrix}$
Test 2	$\begin{bmatrix}10° & 10° & 10°\end{bmatrix}$
Test 3	$\begin{bmatrix}30° & 30° & 30°\end{bmatrix}$
Test 4	$\begin{bmatrix}0° & 90° & 0°\end{bmatrix}$

Table 5.1: Initial attitude estimation errors in Euler angles

5.0.2 Simulation Setup

As no physical test setup was available during the project period, the estimation experiments will be conducted in simulation.

Parameters

Each experiment was performed with the same simulation parameters, that is a simulation time of 120 minutes, with a timestep of 0.1 seconds.

The satellite's orbit was assumed circular and periodic as described in section 2.6

The different sensors' noise variances were based on datasheets primarily from Gomspace/Space Inventor. The star tracker was modeled with extra noise on the

boresight axis, with a standard deviation of 10 arcsec for the boresight axis and 1.5 arcsec for the other two axes, with the covariance matrix in radians:[22]

$$\Sigma_{ST} = \begin{bmatrix} 7.2722\text{e}{-06}^2 & 0 & 0 \\ 0 & 7.2722\text{e}{-06}^2 & 0 \\ 0 & 0 & 4.8481\text{e}{-05}^2 \end{bmatrix} \tag{5.1}$$

The remaining sensors were modelled with isotropic noise, with the fine sun sensor having a standard deviation of 2°, with the covariance matrix in radians:[23]

$$\Sigma_{FSS} = \begin{bmatrix} 0.0349^2 & 0 & 0 \\ 0 & 0.0349^2 & 0 \\ 0 & 0 & 0.0349^2 \end{bmatrix} \tag{5.2}$$

The magnetometer was modelled with a standard deviation of 15 nT, with the covariance matrix in mG:[24]

$$\Sigma_{MAG} = \begin{bmatrix} 0.15^2 & 0 & 0 \\ 0 & 0.15^2 & 0 \\ 0 & 0 & 0.15^2 \end{bmatrix} \tag{5.3}$$

and the gyroscope was modelled with a standard deviation of $0.0017°/\text{s}/\sqrt{\text{Hz}}$, with the covariance matrix in radians:[25]

$$\Sigma_{GYR} = \begin{bmatrix} 2.9671\text{e}{-05}^2 & 0 & 0 \\ 0 & 2.9671\text{e}{-05}^2 & 0 \\ 0 & 0 & 2.9671\text{e}{-05}^2 \end{bmatrix} \tag{5.4}$$

The sensors covariance matrices were used to construct the filters full measurement covariance matrix as:

$$R = \text{blkdiag}(\Sigma_{ST}, \Sigma_{MAG}, \Sigma_{FSS}) \tag{5.5}$$

which at each iteration of the filter, is used to update a local R that contains only the corresponding and currently available measurements.

The gyroscope in Eq. 2.7a was modelled with an initial bias of 0.1 °/h which develops according to a random walk as in Eq. 2.7b, and is also modelled with the following scaling factors and misalignments:

$$b_{init}^{true} = 1.0\text{e}{-06} \begin{bmatrix} 0.4848 \\ 0.4849 \\ 0.4849 \end{bmatrix} \text{ rad/s} \tag{5.6a}$$

$$S^{true} = 1.0\text{e}{-2} \begin{bmatrix} 0.1500 & 0.1000 & 0.1500 \\ 0.0500 & 0.1000 & 0.2000 \\ 0.1000 & 0.1500 & 0.1500 \end{bmatrix} \tag{5.6b}$$

The magnetometer in Eq. 2.12 was modelled with a constant bias along with the following scaling factors:

$$b^{true} = \begin{bmatrix} 50 \\ 60 \\ 55 \end{bmatrix} \text{ mG} \tag{5.7a}$$

$$D^{true} = \begin{bmatrix} 0.0800 & 0.0520 & 0.0500 \\ 0.0520 & 0.0500 & 0.0490 \\ 0.0500 & 0.0490 & 0.0750 \end{bmatrix} \tag{5.7b}$$

These values are estimated in the beginning of the next chapter, and the estimated values will then be used to perform calibration compensation in the mission mode filters.

Chapter 6

Results

In this chapter, the results obtained from the experiments described in chapter 5 are presented. They are further discussed in chapter 7.

6.1 Calibration Filters

6.1.1 Magnetometer EKF

The calibration parameters to be estimated is the following state vector:

$$\hat{x} = \begin{bmatrix} \hat{\beta}_1 & \hat{\beta}_2 & \hat{\beta}_3 & \hat{D}_{11} & \hat{D}_{22} & \hat{D}_{33} & \hat{D}_{12} & \hat{D}_{13} & \hat{D}_{23} \end{bmatrix}^T \tag{6.1}$$

and the resulting estimates are displayed in Fig. 6.1.

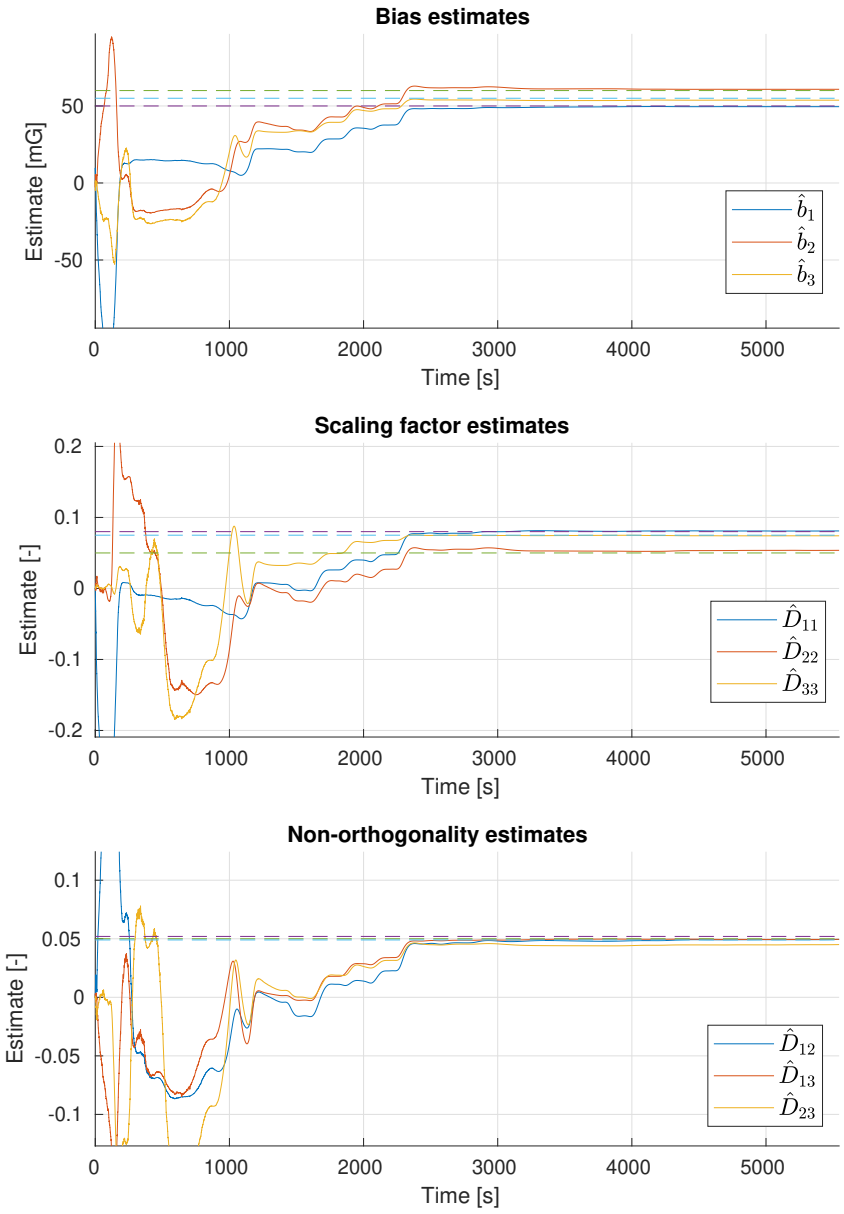

Figure 6.1: Magnetometer calibration EKF results. The filter has large initial deviations but the estimates converge towards the true values displayed with the stapled lines.

The resulting estimated bias vector and scaling matrix converges to:

$$\hat{\beta} = \begin{bmatrix} 49.5309 \\ 60.8206 \\ 53.7601 \end{bmatrix} \tag{6.2a}$$

$$\hat{D} = \begin{bmatrix} 0.0810 & 0.0494 & 0.0495 \\ 0.0494 & 0.0537 & 0.0450 \\ 0.0495 & 0.0450 & 0.0741 \end{bmatrix} \tag{6.2b}$$

The deviation in percentage, of the true states in Eqs. 5.7 from the estimated states $\hat{x}$ is then:

$$\begin{bmatrix} -0.9383\% \\ 1.3677\% \\ -2.2544\% \\ 1.2425\% \\ 7.4128\% \\ -1.1440\% \\ -4.9471\% \\ -0.9564\% \\ -8.0889\% \end{bmatrix} \tag{6.3}$$

The average accuracy of the filter can then be calculated as

$$\text{Accuracy} = \text{mean}(100 - \text{abs}(\text{Deviation})) = 96.8498\% \tag{6.4}$$

6.1.2 Magnetometer UKF

In the UKF calibration the same state vector is estimated:

$$\hat{x} = \begin{bmatrix} \hat{\beta}_1 & \hat{\beta}_2 & \hat{\beta}_3 & \hat{D}_{11} & \hat{D}_{22} & \hat{D}_{33} & \hat{D}_{12} & \hat{D}_{13} & \hat{D}_{23} \end{bmatrix}^T \tag{6.5}$$

and the resulting estimates are displayed in Fig. 6.2.

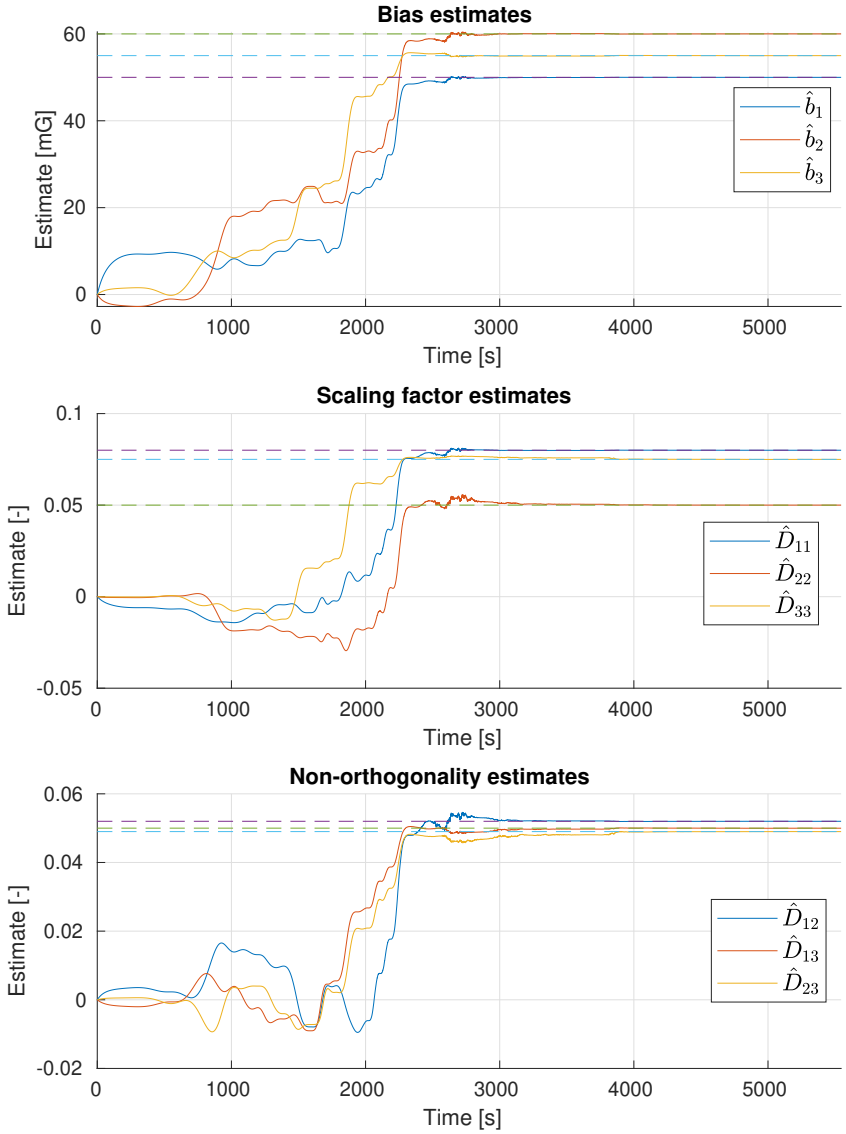

Figure 6.2: Magnetometer calibration UKF results. The filter estimates converge towards the true values with significantly smaller deviations than the EKF.

The estimated bias vector and scaling factor matrix converges to:

$$\hat{\beta} = \begin{bmatrix} 49.9950 \\ 60.0047 \\ 54.9944 \end{bmatrix} \tag{6.6a}$$

$$\hat{D} = \begin{bmatrix} 0.0799 & 0.0520 & 0.0499 \\ 0.0520 & 0.0500 & 0.0489 \\ 0.0499 & 0.0489 & 0.0749 \end{bmatrix} \tag{6.6b}$$

The deviation in percentage, of the true states from the estimated state vector $\hat{x}$ then becomes:

$$\begin{bmatrix} -0.0098\% \\ 0.0078\% \\ -0.0101\% \\ -0.0198\% \\ 0.0106\% \\ -0.0291\% \\ 0.0027\% \\ -0.0266\% \\ -0.0116\% \end{bmatrix} \tag{6.7}$$

with an average estimation accuracy of 99.9857%.

6.1.3 Gyroscope MEKF

In the gyroscope calibration filter we are estimated another 3 parameters, as the scaling factor matrix is no longer symmetrical:

$$\hat{x} = \begin{bmatrix} \hat{\beta}_1 & \hat{\beta}_2 & \hat{\beta}_3 & \hat{S}_{11} & \hat{S}_{22} & \hat{S}_{33} & \hat{S}_{12} & \hat{S}_{13} & \hat{S}_{23} & \hat{S}_{21} & \hat{S}_{31} & \hat{S}_{32} \end{bmatrix}^T \tag{6.8}$$

and the resulting estimates of S are displayed in Fig. 6.3. Plots of $\hat{\beta}$ are left out, as β is continuously estimated when using mission mode filters as well.

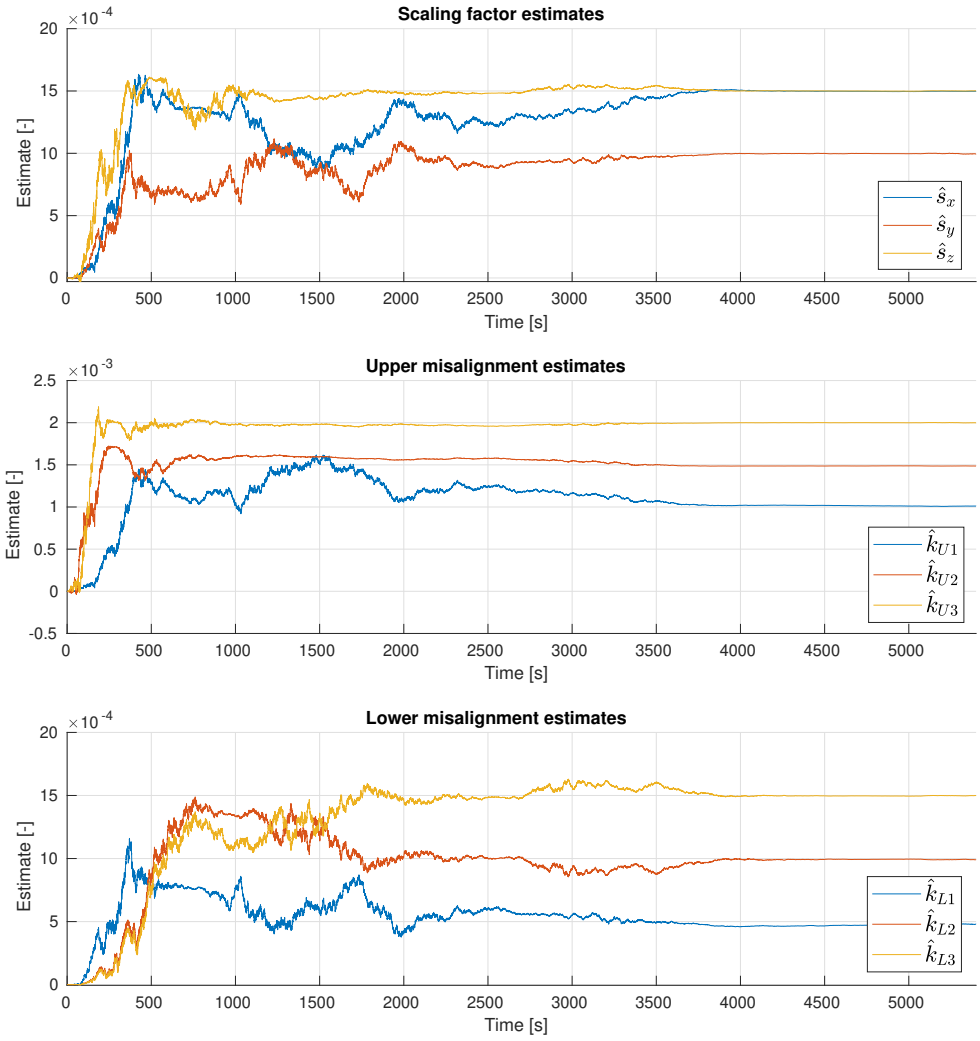

Figure 6.3: Gyroscope calibration MEKF results. The filter estimates, $\hat{S}$, are all seen to converge.

The resulting estimated bias vector and scaling factor matrix are:

$$\hat{\beta} = 1.0\text{e}{-06} \begin{bmatrix} 0.4837 \\ 0.5092 \\ 0.5000 \end{bmatrix} \tag{6.9a}$$

$$\hat{S} = 1.0\text{e}{-2} \begin{bmatrix} 0.1498 & 0.1011 & 0.1486 \\ 0.0479 & 0.0994 & 0.1999 \\ 0.0990 & 0.1500 & 0.1502 \end{bmatrix} \tag{6.9b}$$

which compared to the true bias and scaling factors in Eqs. 5.6, results in the following percentage deviation:

$$\begin{bmatrix} -1.0709\% \\ 1.8918\% \\ -0.3550\% \\ -0.1213\% \\ -0.5429\% \\ 0.1105\% \\ 1.0770\% \\ -0.9336\% \\ -0.0718\% \\ -4.0943\% \\ -0.9897\% \\ -0.0103\% \end{bmatrix} \tag{6.10}$$

which results in an average estimation accuracy of 99.0609%

6.1.4 Smoothed Gyroscope MEKF

The estimated values from the gyroscope MEKF are smoothed as described in section 4.2.1, and the corresponding smoothed estimates are displayed in Fig. 6.4

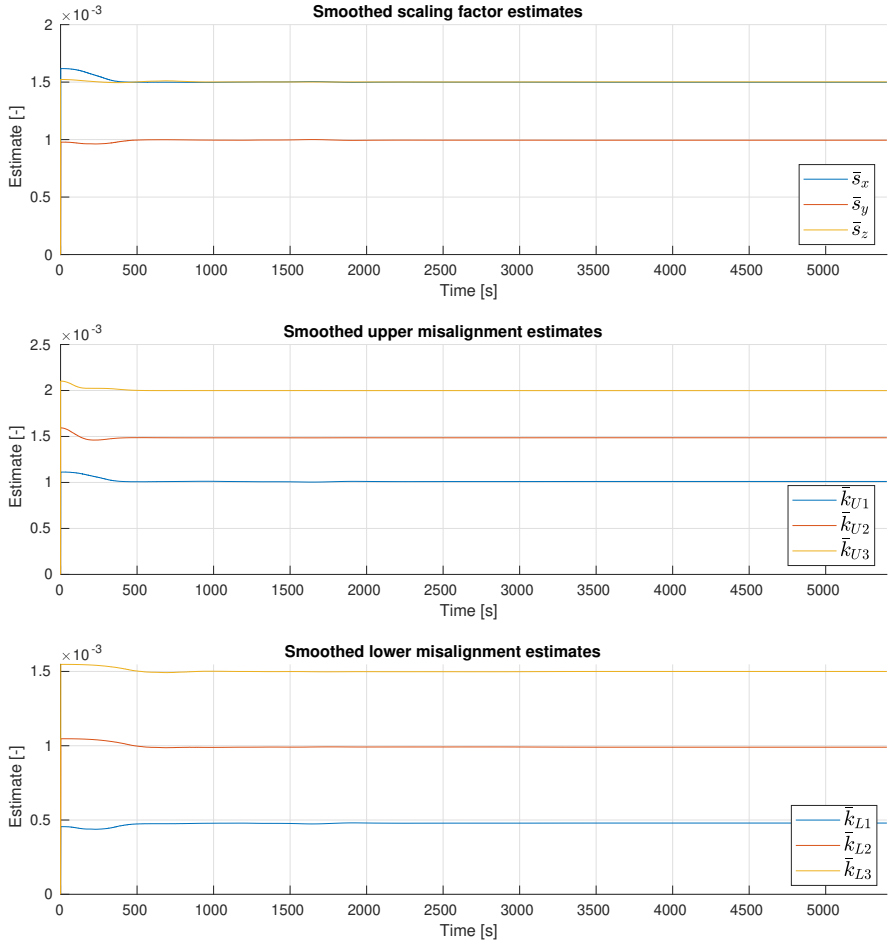

Figure 6.4: Smoothed gyroscope calibration MEKF results.

By taking a mean of the last half of the smoothed values, the estimated bias vector and scaling factor matrix becomes:

$$\hat{\beta} = 1.0\text{e}{-06} \begin{bmatrix} 0.4819 \\ 0.5139 \\ 0.4924 \end{bmatrix} \tag{6.11a}$$

$$\hat{S} = 1.0\text{e}{-2} \begin{bmatrix} 0.1498 & 0.1010 & 0.1485 \\ 0.0479 & 0.0994 & 0.9985 \\ 0.0990 & 0.1499 & 0.1501 \end{bmatrix} \tag{6.11b}$$

which compared to the true bias and scaling factors in Eqs. 5.6, results in the

following percentage deviation:

$$\begin{bmatrix} -1.4501\% \\ 2.8473\% \\ -1.8675\% \\ -0.1072\% \\ -0.5348\% \\ 0.1028\% \\ 1.0600\% \\ -0.9377\% \\ -0.0704\% \\ -4.1161\% \\ -0.9589\% \\ -0.0297\% \end{bmatrix} \tag{6.12}$$

which results in an average estimation accuracy of 98.8265%. It is remarked that this is a slightly lower accuracy than by choosing the final estimate from the non-smoothed series.

6.2 Mission Mode Filters

The following are the results for the mission mode filters simulated under different scenarios as describe in the section 5.0.1. The magnetometer and gyroscope calibration was performed using the estimated values from section 6.1.1 and 6.1.4, to be able to evaluate the performance under a worst case scenario.

6.2.1 Scenario 1

In this scenario, only a star tracker and a gyroscope is available.

Initial 1° Error

Results are summarized in Tab. 6.1.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 2.9e{-}5 \\ 4.3e{-}6 \\ 4.5e{-}6 \end{bmatrix}$	$\begin{bmatrix} 2.9e{-}5 \\ 4.3e{-}6 \\ 4.5e{-}6 \end{bmatrix}$	$\begin{bmatrix} 2.8e{-}5 \\ 4.0e{-}6 \\ 4.2e{-}6 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 0.2137 \\ 0.2548 \\ 0.4221 \end{bmatrix}$	$\begin{bmatrix} 0.2137 \\ 0.2548 \\ 0.4221 \end{bmatrix}$	$\begin{bmatrix} 0.1905 \\ 0.2430 \\ 0.4044 \end{bmatrix}$
Compute Time (s)	45.19	22.77	25.84

Table 6.1: Results of running scenario 1 with the initial error of 1°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

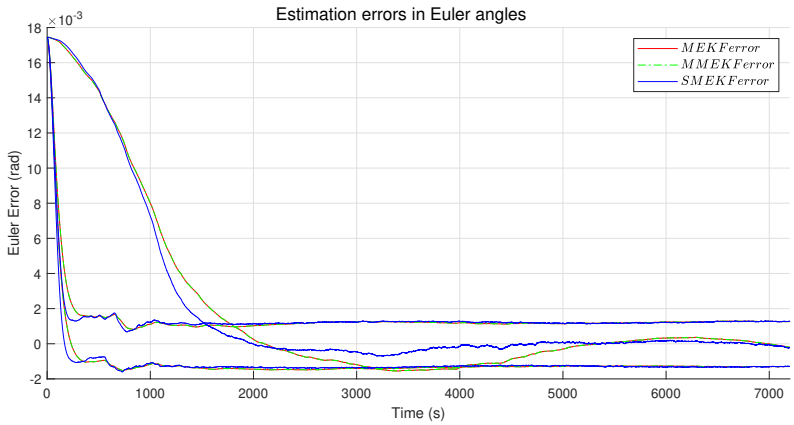

Figure 6.5: Euler Angle error with initial error of 1°

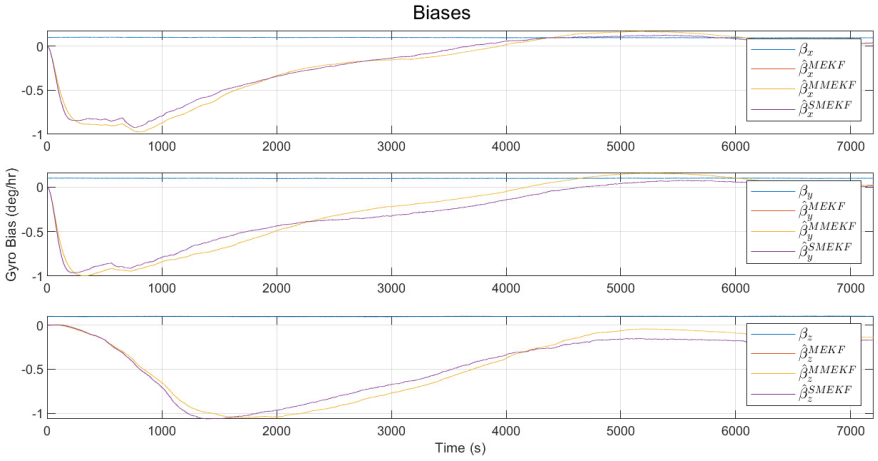

Figure 6.6: Gyro Bias Estimate for initial error of 1°

Initial 10° Error

Results are summarized in Tab. 6.2.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 2.6e{-}3 \\ 2.9e{-}4 \\ 2.9e{-}4 \end{bmatrix}$	$\begin{bmatrix} 2.6e{-}3 \\ 2.9e{-}4 \\ 2.9e{-}4 \end{bmatrix}$	$\begin{bmatrix} 2.5e{-}3 \\ 2.6e{-}4 \\ 2.6e{-}4 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 16.3549 \\ 25.9444 \\ 26.6010 \end{bmatrix}$	$\begin{bmatrix} 16.3527 \\ 25.9414 \\ 26.6003 \end{bmatrix}$	$\begin{bmatrix} 14.2570 \\ 25.0424 \\ 25.0970 \end{bmatrix}$
Compute Time (s)	71.14	27.70	27.60

Table 6.2: Results of running scenario 1 with the initial error of 10°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

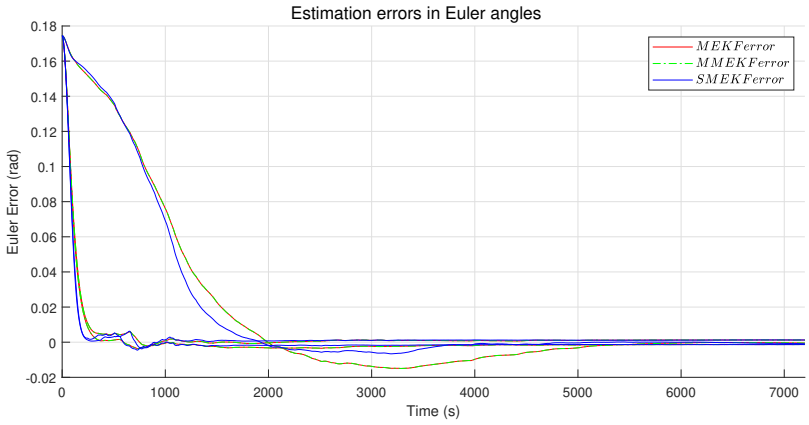

Figure 6.7: Euler Angle error with initial error of 10°

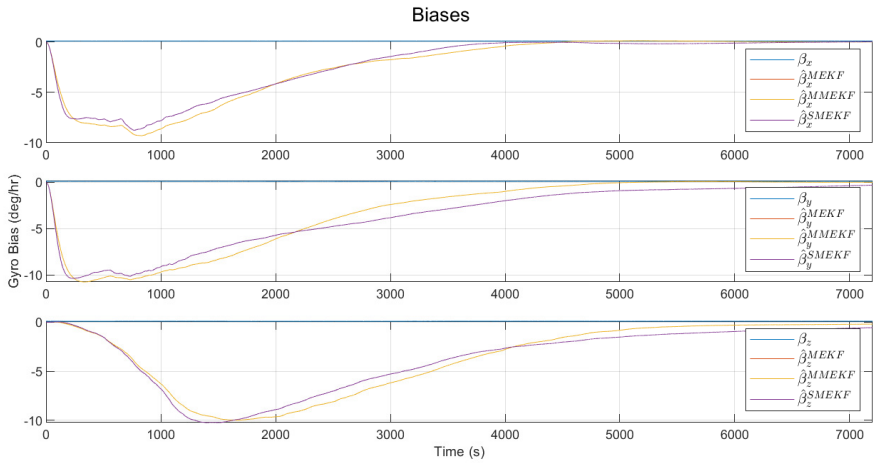

Figure 6.8: Gyro Bias Estimate for initial error of 10°

Initial 30° Error

Results are summarized in Tab. 6.3.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0165 \\ 0.0029 \\ 0.0027 \end{bmatrix}$	$\begin{bmatrix} 0.0165 \\ 0.0029 \\ 0.0027 \end{bmatrix}$	$\begin{bmatrix} 0.0157 \\ 0.0027 \\ 0.0025 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 61.2684 \\ 276.1572 \\ 146.9004 \end{bmatrix}$	$\begin{bmatrix} 61.2607 \\ 276.1237 \\ 146.8949 \end{bmatrix}$	$\begin{bmatrix} 53.1412 \\ 266.2671 \\ 134.8363 \end{bmatrix}$
Compute Time (s)	51.34	22.60	24.34

Table 6.3: Results of running scenario 1 with the initial error of 30°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

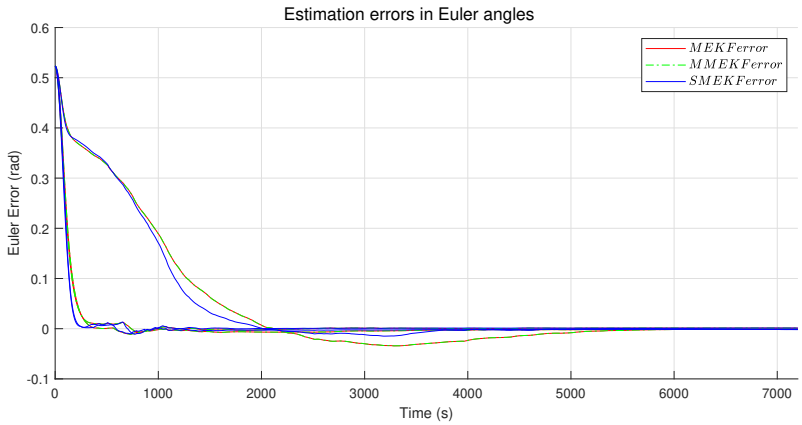

Figure 6.9: Euler Angle error with initial error of 30°

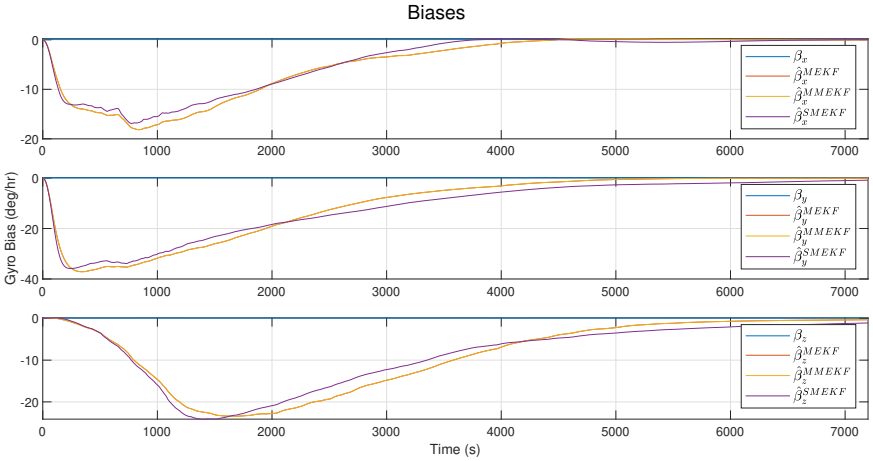

Figure 6.10: Gyro Bias Estimate for initial error of 30°

Initial 90° Error

Results are summarized in Tab. 6.4.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0338 \\ 0.0754 \\ 0.0321 \end{bmatrix}$	$\begin{bmatrix} 0.0338 \\ 0.0754 \\ 0.0321 \end{bmatrix}$	$\begin{bmatrix} 0.0340 \\ 0.0733 \\ 0.0322 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 1.1187 \\ 1.46e3 \\ 8.2170 \end{bmatrix}$	$\begin{bmatrix} 1.1186 \\ 1.46e3 \\ 8.2146 \end{bmatrix}$	$\begin{bmatrix} 2.6964 \\ 1.47e3 \\ 4.2251 \end{bmatrix}$
Compute Time (s)	54.25	22.85	25.16

Table 6.4: Results of running scenario 1 with the initial error of 90°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

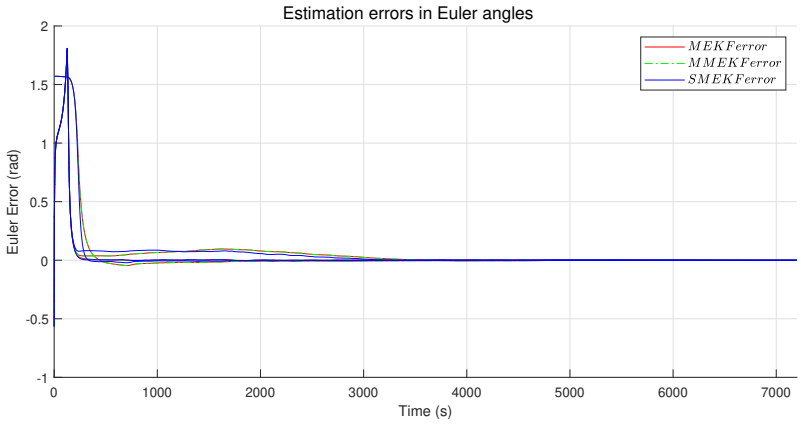

Figure 6.11: Euler Angle error with initial error of 90°

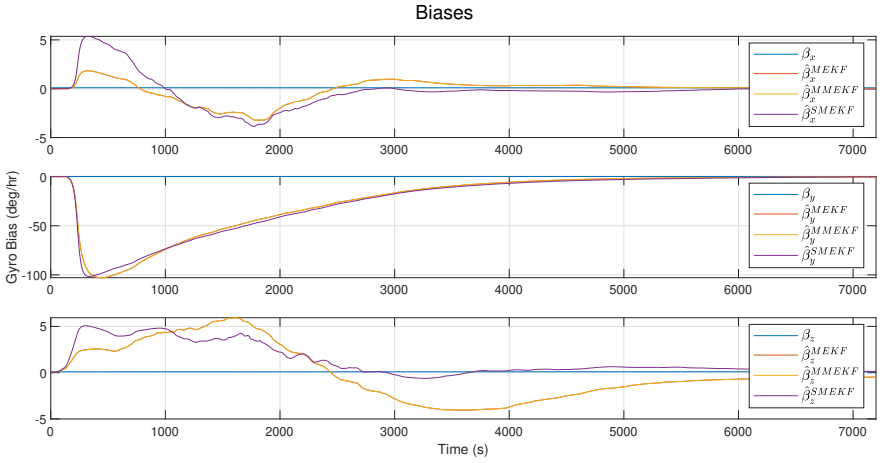

Figure 6.12: Gyro Bias Estimate for initial error of 90°

As a star tracker can track many stars accurately with low noise, the filter response is quick and it converges to the true values with high accuracy even when the initial error is 90°.

6.2.2 Scenario 2

In this scenario, only magnetometers, a gyroscope and fine sun sensors are available.

Initial 1° Error

Results are summarized in Tab. 6.5.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 5.2e{-}5 \\ 1.2e{-}4 \\ 1.8e{-}4 \end{bmatrix}$	$\begin{bmatrix} 5.2e{-}5 \\ 1.2e{-}4 \\ 1.8e{-}4 \end{bmatrix}$	$\begin{bmatrix} 6.2e{-}5 \\ 2.5e{-}5 \\ 4.2e{-}4 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 8.894 \\ 7.374 \\ 13.331 \end{bmatrix}$	$\begin{bmatrix} 8.894 \\ 7.374 \\ 13.331 \end{bmatrix}$	$\begin{bmatrix} 7.664 \\ 5.482 \\ 15.3242 \end{bmatrix}$
Compute Time (s)	50.78	19.63	19.88

Table 6.5: Results of running scenario 2 with the initial error of 1°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

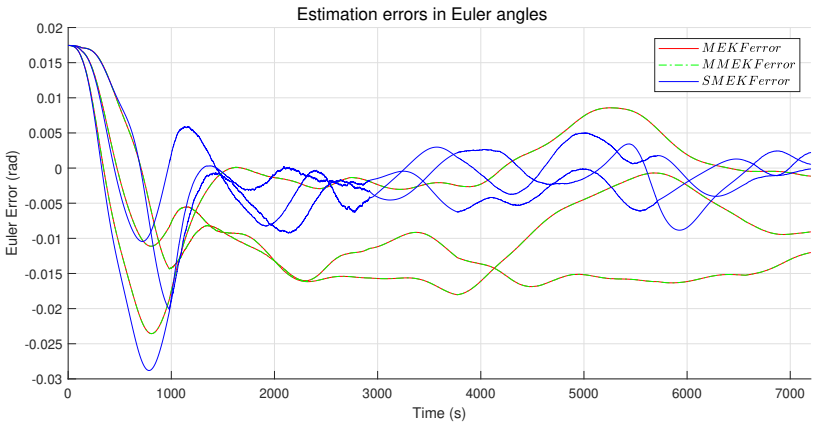

Figure 6.13: Euler Angle error with initial error of 1°

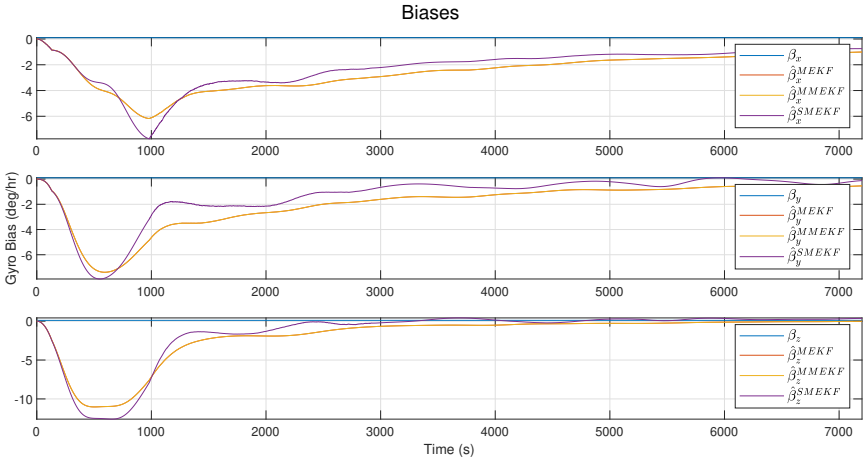

Figure 6.14: Gyro Bias Estimate for initial error of 1°

Initial 10° Error

Results are summarized in Tab. 6.6.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 3.2e{-}3 \\ 8.3e{-}3 \\ 1.3e{-}2 \end{bmatrix}$	$\begin{bmatrix} 3.2e{-}3 \\ 8.3e{-}3 \\ 1.3e{-}2 \end{bmatrix}$	$\begin{bmatrix} 4.0e{-}3 \\ 1.7e{-}3 \\ 3.1e{-}3 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 812.30 \\ 594.93 \\ 985.10 \end{bmatrix}$	$\begin{bmatrix} 812.30 \\ 594.92 \\ 985.15 \end{bmatrix}$	$\begin{bmatrix} 654.88 \\ 413.36 \\ 1168.8 \end{bmatrix}$
Compute Time (s)	38.59	15.04	15.76

Table 6.6: Results of running scenario 2 with the initial error of 10°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

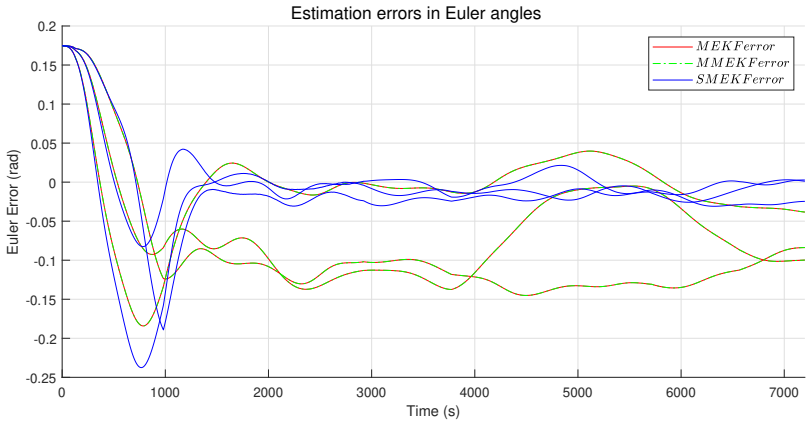

Figure 6.15: Euler Angle error with initial error of 10°

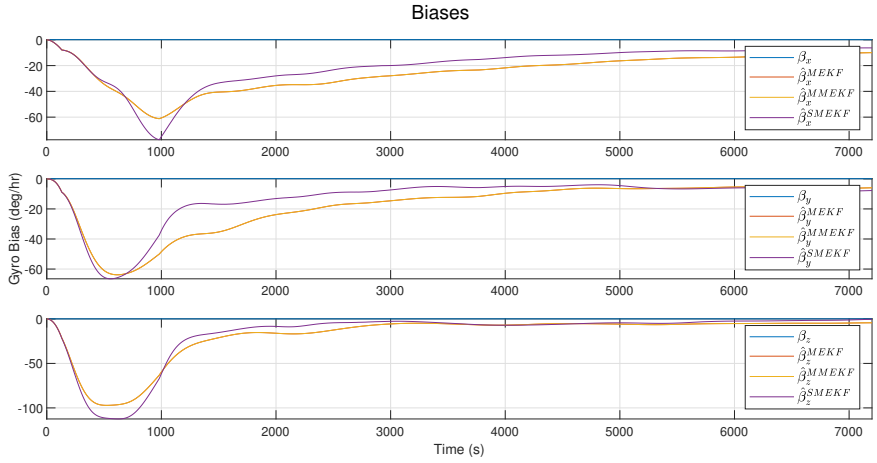

Figure 6.16: Gyro Bias Estimate for initial error of 10°

Initial 30° Error

Results are summarized in Tab. 6.7.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 2.8e{-}2 \\ 7.6e{-}2 \\ 1.3e{-}2 \end{bmatrix}$	$\begin{bmatrix} 2.8e{-}2 \\ 7.6e{-}2 \\ 1.3e{-}2 \end{bmatrix}$	$\begin{bmatrix} 3.1e{-}5 \\ 1.6e{-}2 \\ 2.3e{-}2 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 5.9e3 \\ 5.7e3 \\ 6.4e3 \end{bmatrix}$	$\begin{bmatrix} 5.9e3 \\ 5.7e3 \\ 6.4e3 \end{bmatrix}$	$\begin{bmatrix} 4.2e3 \\ 3.9e3 \\ 8.4e3 \end{bmatrix}$
Compute Time (s)	39.5	15.28	15.86

Table 6.7: Results of running scenario 2 with the initial error of 30°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

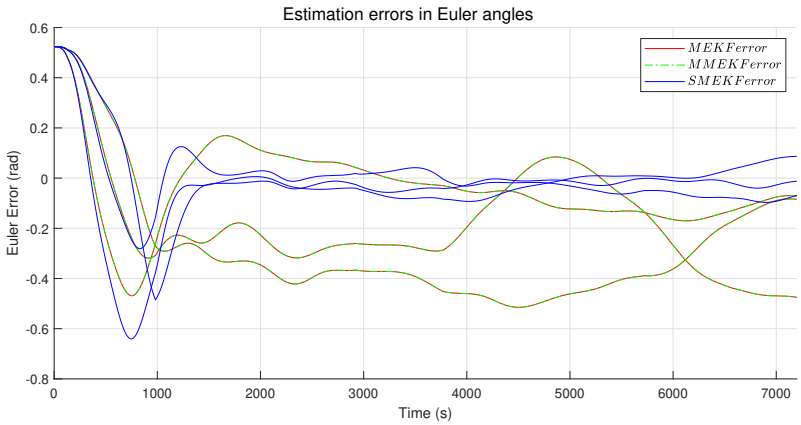

Figure 6.17: Euler Angle error with initial error of 30°

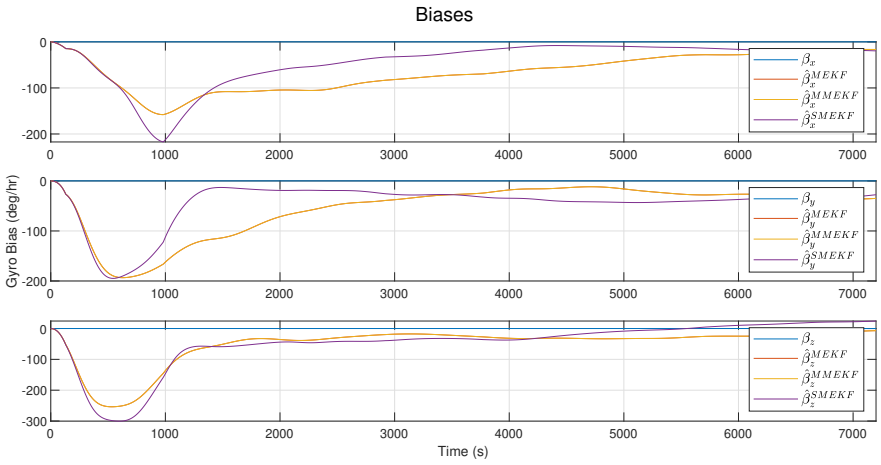

Figure 6.18: Gyro Bias Estimate for initial error of 30°

Initial 90° Error

Results are summarized in Tab. 6.8.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.3901 \\ 0.3375 \\ 0.5472 \end{bmatrix}$	$\begin{bmatrix} 0.3900 \\ 0.3375 \\ 0.5470 \end{bmatrix}$	$\begin{bmatrix} 0.0594 \\ 0.1223 \\ 0.0463 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 5.2e3 \\ 5.5e4 \\ 6.0e3 \end{bmatrix}$	$\begin{bmatrix} 5.2e3 \\ 5.5e4 \\ 6.0e3 \end{bmatrix}$	$\begin{bmatrix} 4.5e3 \\ 3.2e4 \\ 1.0e4 \end{bmatrix}$
Compute Time (s)	47.51	18.62	19.18

Table 6.8: Results of running scenario 2 with the initial error of 90°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

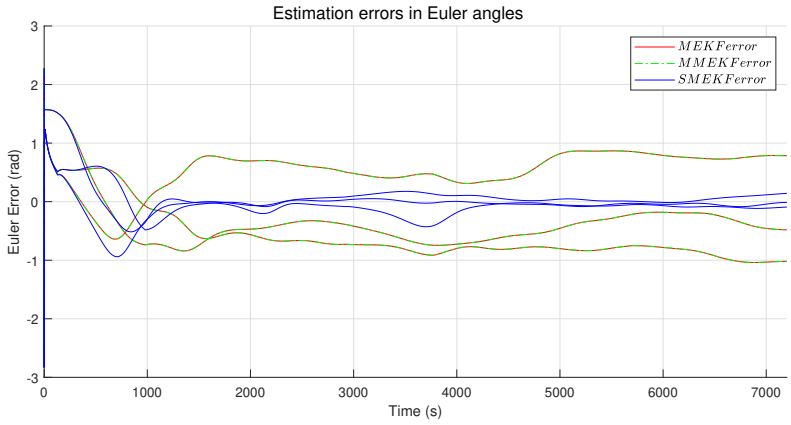

Figure 6.19: Euler Angle error with initial error of 90°

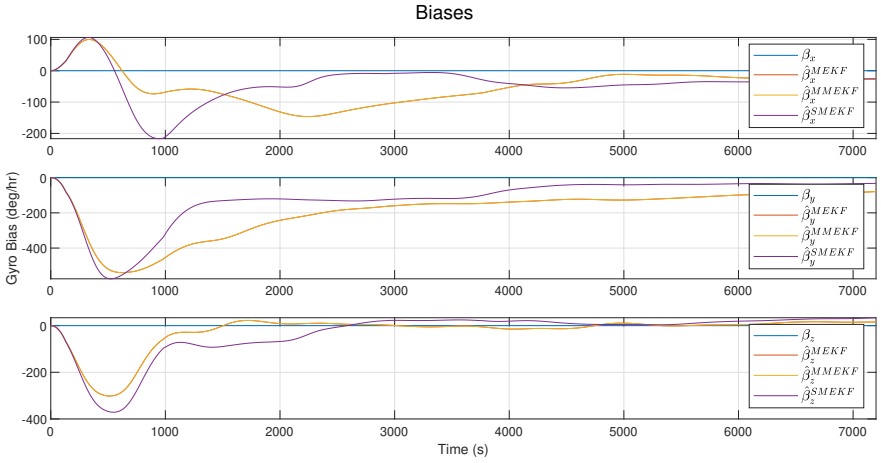

Figure 6.20: Gyro Bias Estimate for initial error of 90°

Both magnetometers and fine sun sensors provide only 3 sets of measurements in the body frame each, compared to the max of 10 from the star tracker. As such the filter converges slower and is less accurate compared to the filter in scenario 1.

6.2.3 Scenario 3

In this scenario only magnetometers and a gyroscope are available.

Initial 1° Error

Results are summarized in Tab. 6.9.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 5.1e{-}5 \\ 1.2e{-}4 \\ 2.1e{-}5 \end{bmatrix}$	$\begin{bmatrix} 5.1e{-}5 \\ 1.2e{-}4 \\ 2.1e{-}5 \end{bmatrix}$	$\begin{bmatrix} 6.3e{-}5 \\ 3.2e{-}5 \\ 7.8e{-}5 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 9.1158 \\ 7.4925 \\ 13.153 \end{bmatrix}$	$\begin{bmatrix} 9.1158 \\ 7.4925 \\ 13.153 \end{bmatrix}$	$\begin{bmatrix} 8.9148 \\ 5.3076 \\ 15.5836 \end{bmatrix}$
Compute Time (s)	22.81	13.22	13.60

Table 6.9: Results of running scenario 3 with the initial error of 1°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

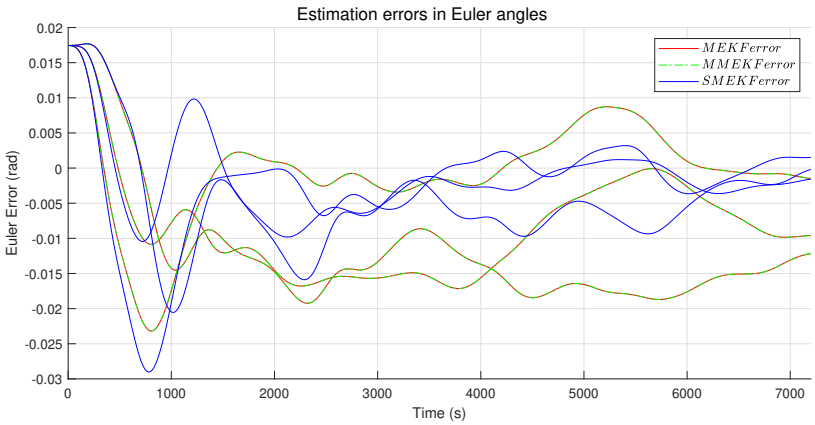

Figure 6.21: Euler Angle error with initial error of 1°

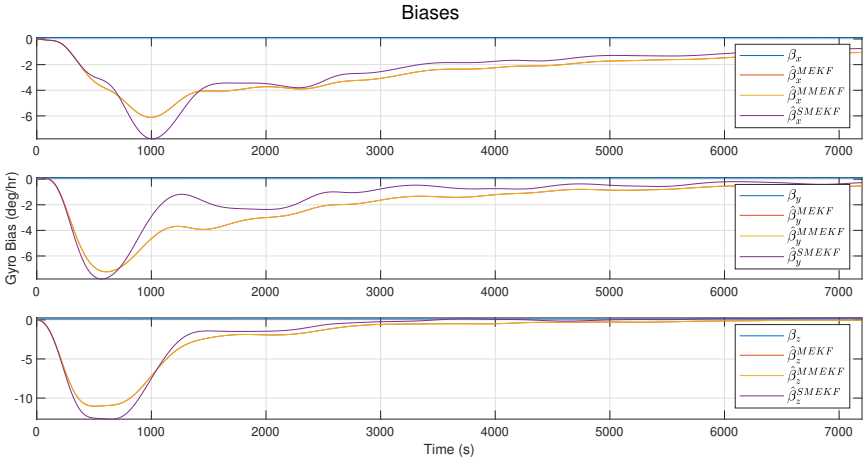

Figure 6.22: Gyro Bias Estimate for initial error of 1°

Initial 10° Error

Results are summarized in Tab. 6.10.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 3.3e{-}3 \\ 8.4e{-}3 \\ 1.5e{-}2 \end{bmatrix}$	$\begin{bmatrix} 3.3e{-}3 \\ 8.4e{-}3 \\ 1.5e{-}2 \end{bmatrix}$	$\begin{bmatrix} 4.4e{-}3 \\ 2.4e{-}3 \\ 4.7e{-}3 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 820.21 \\ 600.59 \\ 976.35 \end{bmatrix}$	$\begin{bmatrix} 820.23 \\ 600.57 \\ 976.41 \end{bmatrix}$	$\begin{bmatrix} 738.06 \\ 385.98 \\ 1210 \end{bmatrix}$
Compute Time (s)	21.77	12.91	13.11

Table 6.10: Results of running scenario 3 with the initial error of 10°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

Figure 6.23: Euler Angle error with initial error of 10°

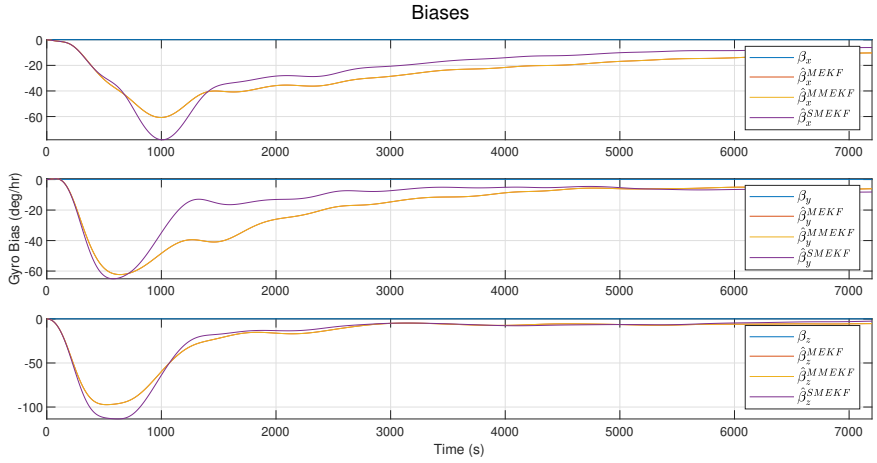

Figure 6.24: Gyro Bias Estimate for initial error of 10°

Initial 30° Error

Results are summarized in Tab. 6.11.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0527 \\ 0.0767 \\ 0.1324 \end{bmatrix}$	$\begin{bmatrix} 0.0527 \\ 0.0767 \\ 0.1324 \end{bmatrix}$	$\begin{bmatrix} 0.0281 \\ 0.0182 \\ 0.0204 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 3.06e3 \\ 3.15e3 \\ 3.35e3 \end{bmatrix}$	$\begin{bmatrix} 3.06e3 \\ 3.15e3 \\ 3.35e3 \end{bmatrix}$	$\begin{bmatrix} 2.77e3 \\ 1.97e3 \\ 4.66e3 \end{bmatrix}$
Compute Time (s)	45.21	25.92	26.44

Table 6.11: Results of running scenario 3 with the initial error of 30°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

Figure 6.25: Euler Angle error with initial error of 30°

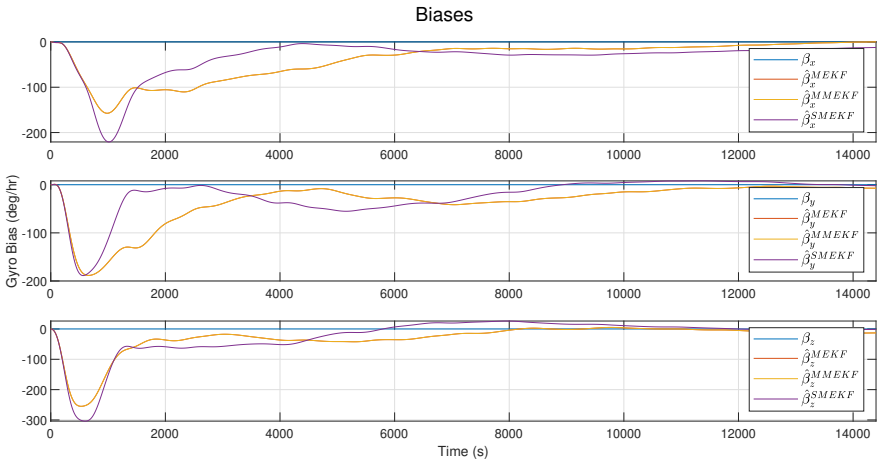

Figure 6.26: Gyro Bias Estimate for initial error of 30°

Initial 90° Error

Results are summarized in Tab. 6.12.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.3197 \\ 0.2333 \\ 0.6055 \end{bmatrix}$	$\begin{bmatrix} 0.3197 \\ 0.2333 \\ 0.6055 \end{bmatrix}$	$\begin{bmatrix} 0.0467 \\ 0.0780 \\ 0.0456 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 2.6e3 \\ 3.14e4 \\ 3.06e3 \end{bmatrix}$	$\begin{bmatrix} 2.6e3 \\ 3.14e4 \\ 3.06e3 \end{bmatrix}$	$\begin{bmatrix} 2.9e3 \\ 1.60e4 \\ 5.15e3 \end{bmatrix}$
Compute Time (s)	43.70	26.192	26.11

Table 6.12: Results of running scenario 3 with the initial error of 90°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

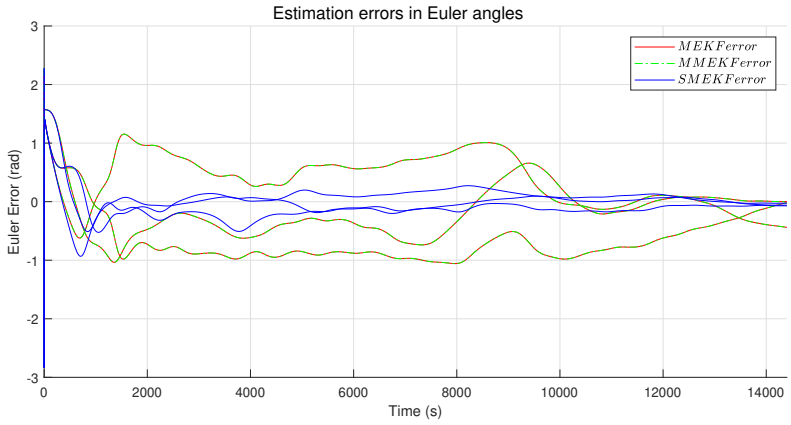

Figure 6.27: Euler Angle error with initial error of 90°

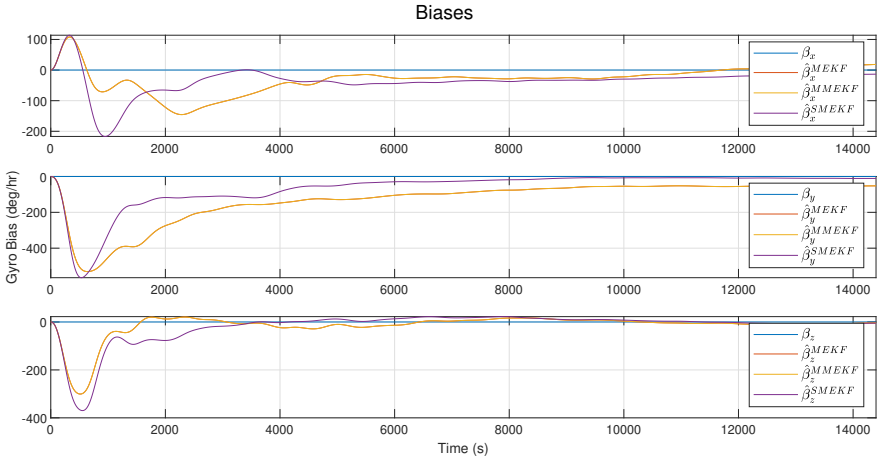

Figure 6.28: Gyro Bias Estimate for initial error of 90°

With the absence of the fine sun sensor, now the filter only has the magnetometer for measurements but these filters still perform only marginally worse than the filters in scenario 2.

6.2.4 Scenario 4

In this scenario, only the fine sun sensors and a gyroscope are available.

Initial 1° Error

Results are summarized in Tab. 6.13.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 1.3e{-}5 \\ 2.4e{-}5 \\ 1.1e{-}4 \end{bmatrix}$	$\begin{bmatrix} 1.3e{-}5 \\ 2.4e{-}5 \\ 1.1e{-}4 \end{bmatrix}$	$\begin{bmatrix} 1.2e{-}5 \\ 2.1e{-}5 \\ 6.9e{-}5 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 0.8544 \\ 0.1726 \\ 0.0570 \end{bmatrix}$	$\begin{bmatrix} 0.8544 \\ 0.1726 \\ 0.0570 \end{bmatrix}$	$\begin{bmatrix} 0.8154 \\ 0.1542 \\ 0.0799 \end{bmatrix}$
Compute Time (s)	46.11	29.23	30.58

Table 6.13: Results of running scenario 4 with the initial error of 1°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

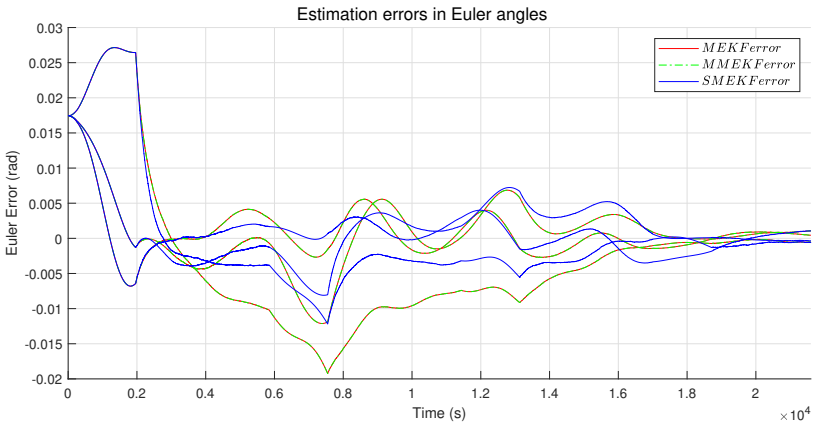

Figure 6.29: Euler Angle error with initial error of 1°

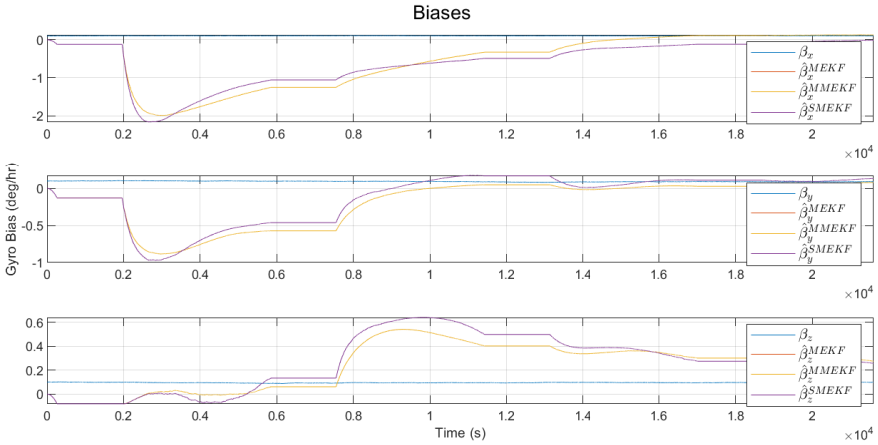

Figure 6.30: Gyro Bias Estimate for initial error of 1°

Initial 10° Error

Results are summarized in Tab. 6.14.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0019 \\ 0.0028 \\ 0.0113 \end{bmatrix}$	$\begin{bmatrix} 0.0019 \\ 0.0028 \\ 0.0113 \end{bmatrix}$	$\begin{bmatrix} 0.0015 \\ 0.0024 \\ 0.0071 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 77.2892 \\ 21.6684 \\ 5.5925 \end{bmatrix}$	$\begin{bmatrix} 77.2892 \\ 21.6684 \\ 5.5925 \end{bmatrix}$	$\begin{bmatrix} 74.8612 \\ 18.6917 \\ 10.0197 \end{bmatrix}$
Compute Time (s)	51.17	31.28	33.17

Table 6.14: Results of running scenario 4 with the initial error of 10°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

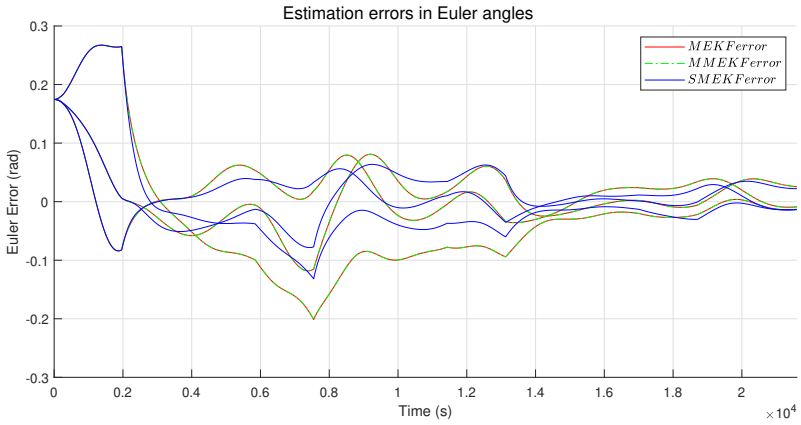

Figure 6.31: Euler Angle error with initial error of 10°

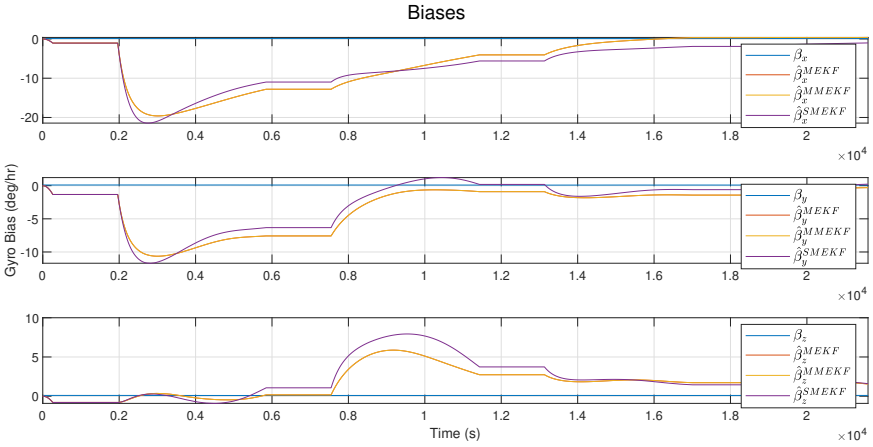

Figure 6.32: Gyro Bias Estimate for initial error of 10°

Initial 30° Error

Results are summarized in Tab. 6.15.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0316 \\ 0.0496 \\ 0.0709 \end{bmatrix}$	$\begin{bmatrix} 0.0316 \\ 0.0496 \\ 0.0709 \end{bmatrix}$	$\begin{bmatrix} 0.0293 \\ 0.0474 \\ 0.0313 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 823.23 \\ 647.42 \\ 157.88 \end{bmatrix}$	$\begin{bmatrix} 823.23 \\ 647.42 \\ 157.88 \end{bmatrix}$	$\begin{bmatrix} 757.68 \\ 529.01 \\ 246.13 \end{bmatrix}$
Compute Time (s)	49.15	30.06	30.96

Table 6.15: Results of running scenario 4 with the initial error of 30°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

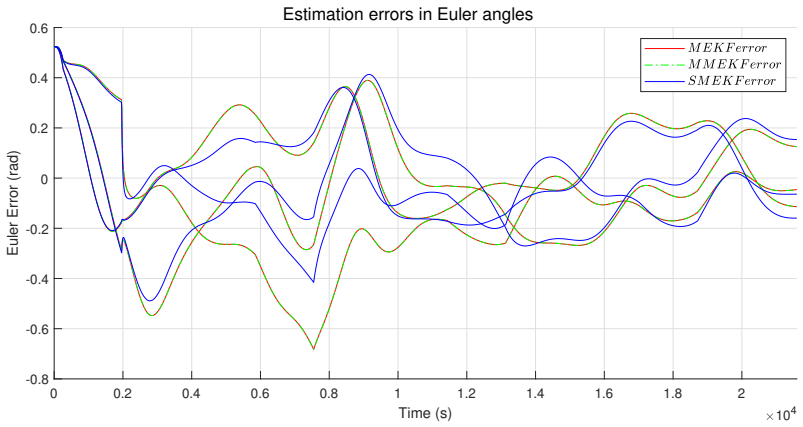

Figure 6.33: Euler Angle error with initial error of 30°

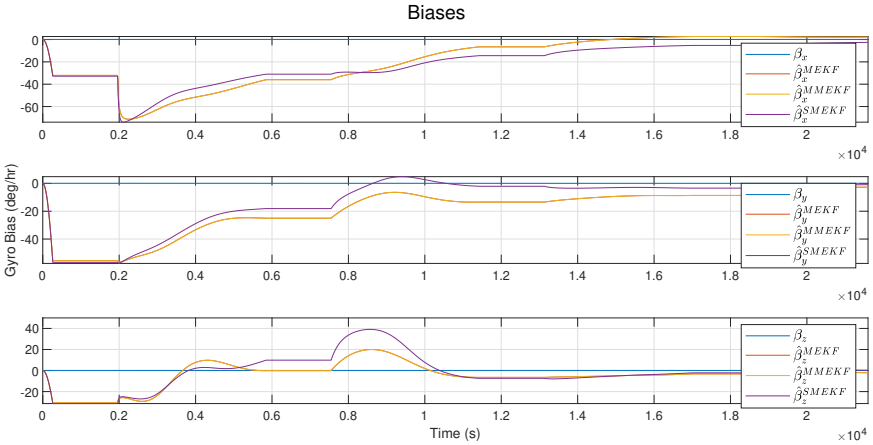

Figure 6.34: Gyro Bias Estimate for initial error of 30°

Initial 90° Error

Results are summarized in Tab. 6.16.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.6768 \\ 0.5801 \\ 0.5529 \end{bmatrix}$	$\begin{bmatrix} 0.6768 \\ 0.5801 \\ 0.5529 \end{bmatrix}$	$\begin{bmatrix} 0.3955 \\ 0.4814 \\ 0.2017 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 1.8e3 \\ 2.7e3 \\ 6.1e2 \end{bmatrix}$	$\begin{bmatrix} 1.8e3 \\ 2.7e3 \\ 6.1e2 \end{bmatrix}$	$\begin{bmatrix} 1.6e3 \\ 1.8e3 \\ 2.0e2 \end{bmatrix}$
Compute Time (s)	50.30	30.23	30.96

Table 6.16: Results of running scenario 4 with the initial error of 90°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

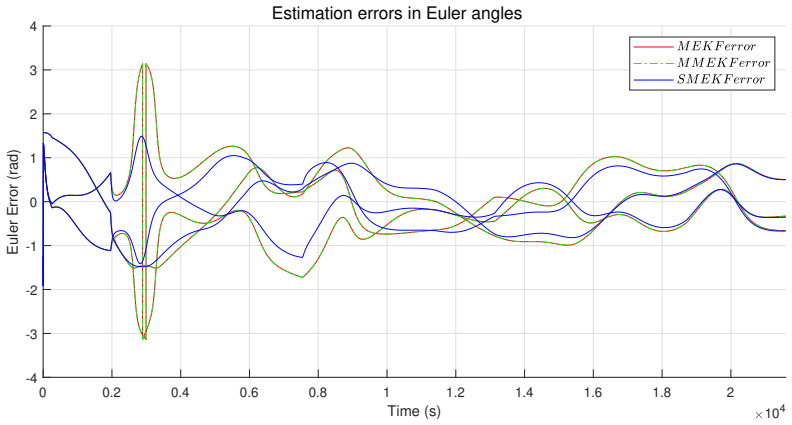

Figure 6.35: Euler Angle error with initial error of 90°

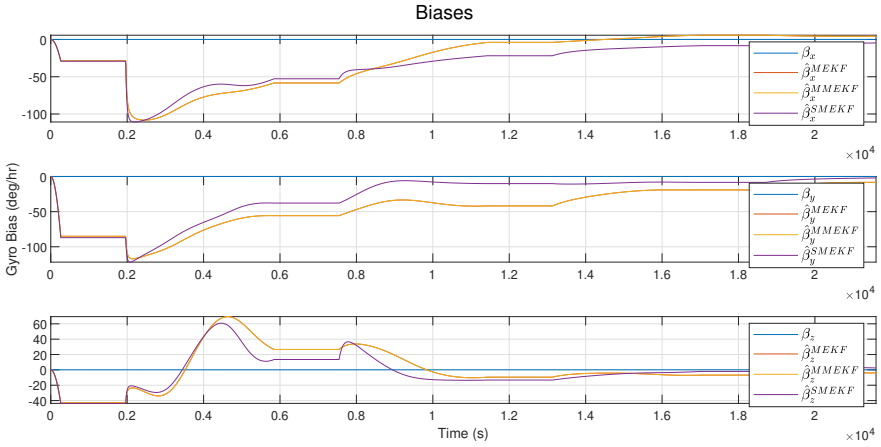

Figure 6.36: Gyro Bias Estimate for initial error of 90°

In this scenario only the fine sun sensor and gyro are available, but the former is also unavailable periodically when the Earth covers the view of the sun. Therefore in this period the filter has no measurements to correct its predictions but still for relatively small initial error the filter does converge with slightly subpar accuracy.

6.2.5 Scenario 5

In this scenario, all the sensors i.e. star tracker, gyroscope, magnetometer and fine sun sensor are available.

Initial 1° Error

Results are summarized in Tab. 6.17.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 2.9e{-5} \\ 4.3e{-6} \\ 4.5e{-6} \end{bmatrix}$	$\begin{bmatrix} 2.9e{-5} \\ 4.3e{-6} \\ 4.5e{-6} \end{bmatrix}$	$\begin{bmatrix} 2.8e{-5} \\ 4.0e{-6} \\ 4.2e{-6} \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 0.2127 \\ 0.2539 \\ 0.4210 \end{bmatrix}$	$\begin{bmatrix} 0.2127 \\ 0.2539 \\ 0.4210 \end{bmatrix}$	$\begin{bmatrix} 0.1896 \\ 0.2419 \\ 0.4031 \end{bmatrix}$
Compute Time (s)	61.59	30.21	33.44

Table 6.17: Results of running scenario 5 with the initial error of 1°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

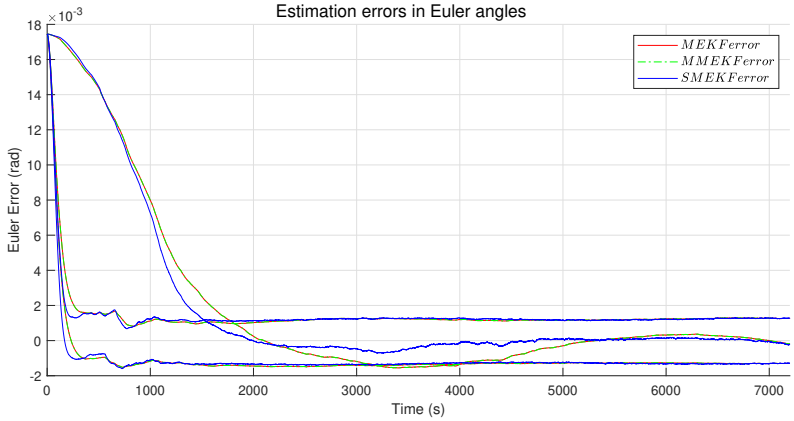

Figure 6.37: Euler Angle error with initial error of 1°

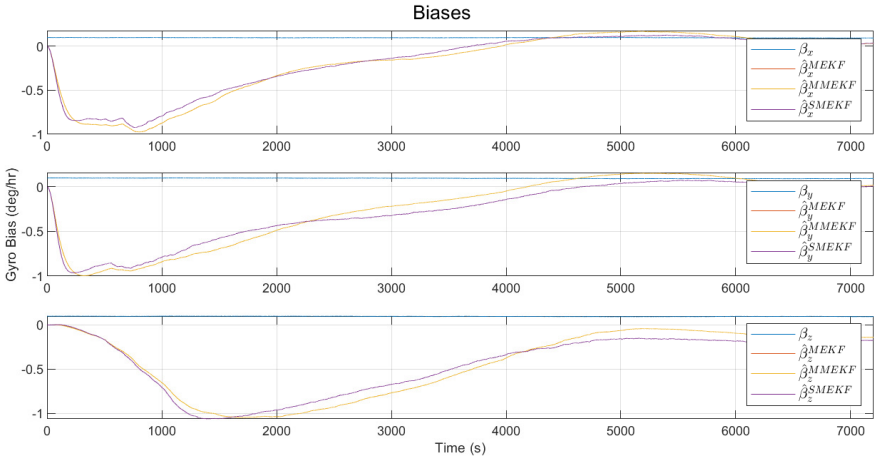

Figure 6.38: Gyro Bias Estimate for initial error of 1°

Initial 10° Error

Results are summarized in Tab. 6.18.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 2.6e{-}3 \\ 2.9e{-}4 \\ 3.0e{-}4 \end{bmatrix}$	$\begin{bmatrix} 2.6e{-}3 \\ 2.9e{-}4 \\ 3.0e{-}4 \end{bmatrix}$	$\begin{bmatrix} 2.5e{-}3 \\ 2.6e{-}4 \\ 2.6e{-}4 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 16.3377 \\ 25.9123 \\ 26.5885 \end{bmatrix}$	$\begin{bmatrix} 16.3377 \\ 25.9123 \\ 26.5885 \end{bmatrix}$	$\begin{bmatrix} 14.2419 \\ 25.0160 \\ 25.0827 \end{bmatrix}$
Compute Time (s)	59.95	31.52	33.32

Table 6.18: Results of running scenario 5 with the initial error of 10°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

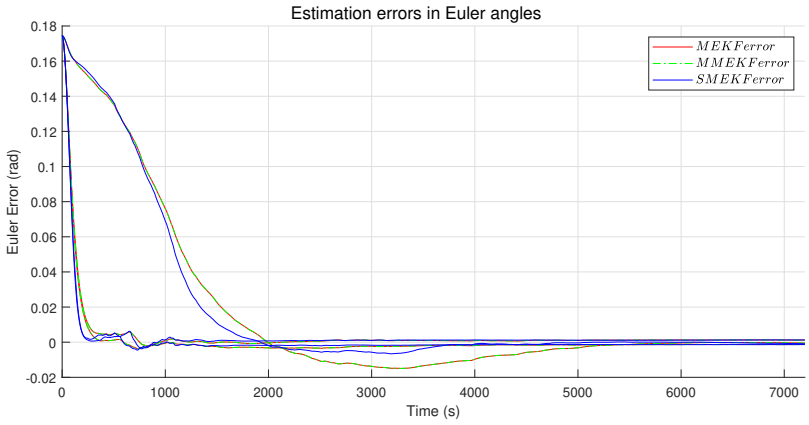

Figure 6.39: Euler Angle Error with initial Error 10°

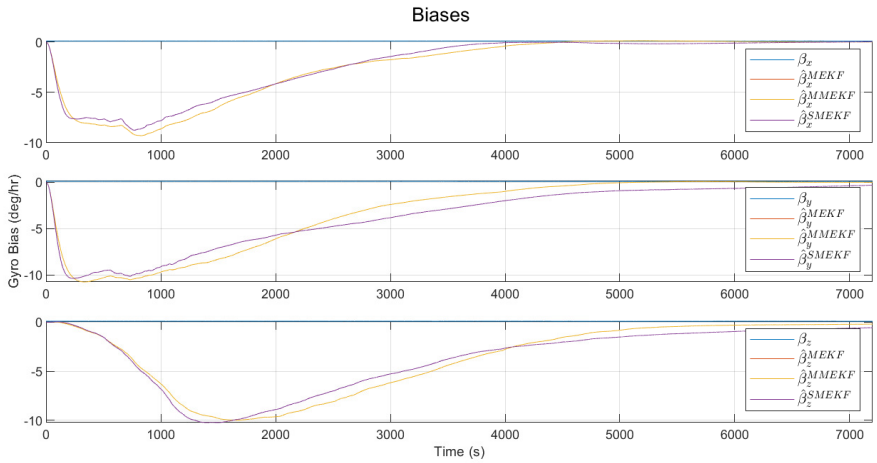

Figure 6.40: Gyro Bias Estimate for initial Error of 10°

Initial 30° Error

Results are summarized in Tab. 6.19.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0165 \\ 0.0029 \\ 0.0027 \end{bmatrix}$	$\begin{bmatrix} 0.0165 \\ 0.0029 \\ 0.0027 \end{bmatrix}$	$\begin{bmatrix} 0.0157 \\ 0.0027 \\ 0.0025 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 61.2533 \\ 276.1189 \\ 146.8834 \end{bmatrix}$	$\begin{bmatrix} 61.2456 \\ 276.0854 \\ 146.8779 \end{bmatrix}$	$\begin{bmatrix} 53.1261 \\ 266.2336 \\ 134.8183 \end{bmatrix}$
Compute Time (s)	64.61	35.40	35.08

Table 6.19: Results of running scenario 5 with the initial error of 30°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

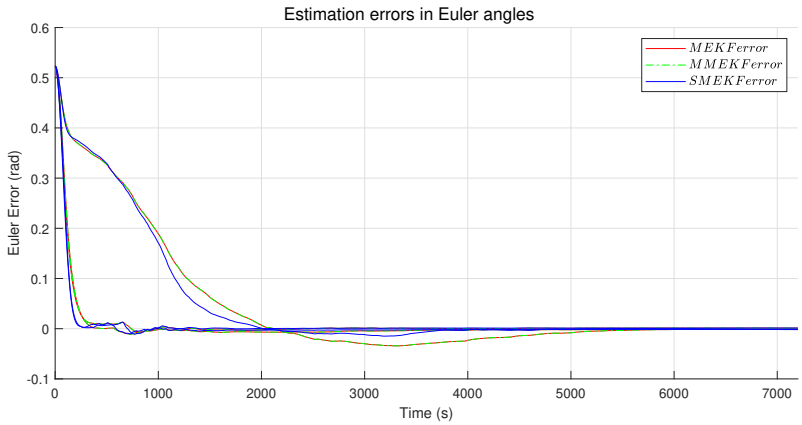

Figure 6.41: Euler Angle error with initial error of 30°

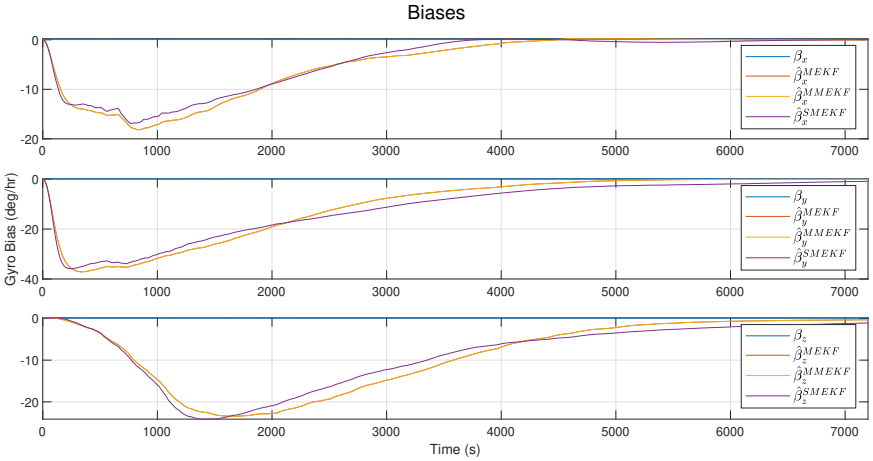

Figure 6.42: Gyro Bias Estimate for initial error of 30°

Initial 90° Error

Results are summarized in Tab. 6.20.

Filter Type	MEKF	MMEKF	SMEKF
Euler Error (MSE)	$\begin{bmatrix} 0.0334 \\ 0.0754 \\ 0.0317 \end{bmatrix}$	$\begin{bmatrix} 0.0334 \\ 0.0753 \\ 0.0317 \end{bmatrix}$	$\begin{bmatrix} 0.0335 \\ 0.0732 \\ 0.0318 \end{bmatrix}$
Gyro Biases (MSE)	$\begin{bmatrix} 1.1171 \\ 1.46e3 \\ 8.2014 \end{bmatrix}$	$\begin{bmatrix} 1.1170 \\ 1.46e3 \\ 8.1981 \end{bmatrix}$	$\begin{bmatrix} 2.6944 \\ 1.47e3 \\ 4.2218 \end{bmatrix}$
Compute Time (s)	63.14	30.86	35.26

Table 6.20: Results of running scenario 5 with the initial error of 90°. The results display MSE for attitude estimation and gyro bias, as well as computation time for the regular MEKF, Murrell's MEKF (MMEKF) and the Sequential MEKF (SMEKF).

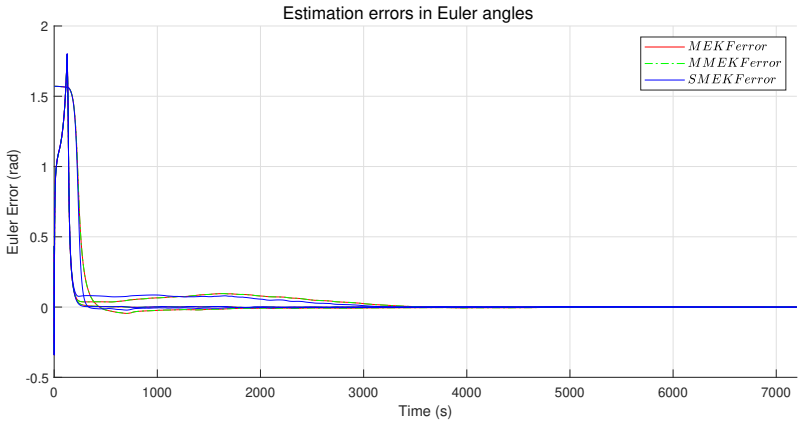

Figure 6.43: Euler Angle error with initial error of 90°

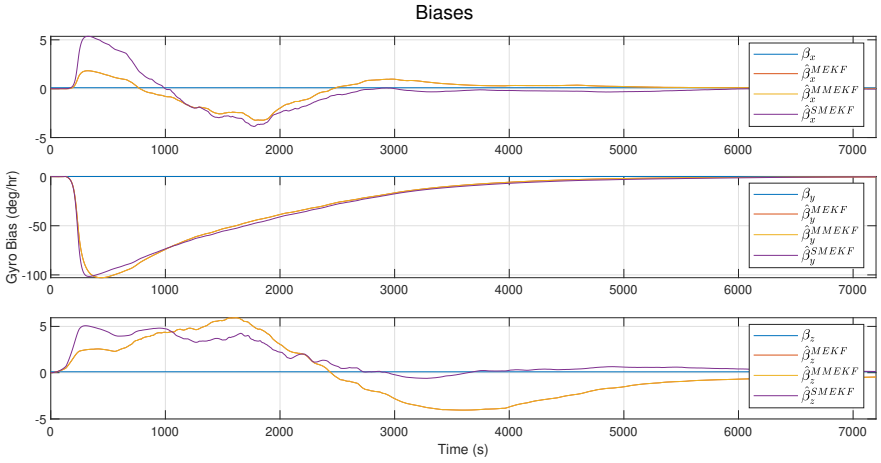

Figure 6.44: Gyro Bias Estimate for initial error of 90°

The filter's response in this scenario where all the sensors are available is practically identical to the one in the 1st scenario. Because of the number of data points and the accuracy a star tracker provides, the contribution of other sensors is almost negligible.

Chapter 7

Discussion and Conclusion

In this chapter, the results from chapter 6 are discussed and a conclusion of the work done during the project is presented.

7.1 Discussion

Calibration

Looking at the results in section 6.1, specifically the deviations and accuracies, it becomes obvious that the UKF performs better than the EKF for magnetometer calibration. This is because it doesn't rely on linearization, but instead propagates the sigma points through the non-linear model. As the model in question is highly non-linear, it is to be expected that the UKF would perform better.

Looking at the accuracies of the smoothed and non-smoothed gyroscope calibration values, it can be seen that the smoothing in this case does not provide an increase in accuracy, but actually leads to a slightly worse result. As the smoothing is a batch process, it cannot be implemented for on-line calibration. Based on the fact that it does not provide an increase in accuracy, and that it can't be implemented in real-time, it does not make sense to use resources on smoothing.

Mission Mode

From the results in section 6.2 it can be seen that the SMEKF is the best filter, as it provides the most accurate estimates without being computationally expensive. This is especially obvious when taking into account larger initial attitude errors. Comparing the other two, it is clearly observed that both the MEKF and MMEKF provide identical results while the former also uses more than double the computational time of the former, making MEKF the worst choice for on-line implementation.

From the different scenarios we can observe that the star tracker performs the best, and if we have a star tracker available, then the addition of magnetometers

and sun sensors does not improve the accuracy and response of the filter. But on the other hand from scenario 2, 3 and 4 it is clear that the combination of sun sensors and magnetometers works better than either of them working on their own. In scenario 4, measurements from the sun sensor were periodically interrupted by the earth. But even in the the absence of any measurements, the filter still converges as long as the initial error is not too large.

Looking at the bias convergence, it can be seen that the convergence time is independent of the initial attitude error. Only the magnitude of the initial bias estimate error, and therefore the MSE, is affected by an increase in initial attitude error.

7.2 Conclusion

In this work several filters for high-precision attitude determination of a spacecraft were investigated. Non-linear estimation methods such as different variations of the Extended Kalman Filter and Unscented Kalman Filter were implemented for on-line attitude estimation and sensor calibration. The experiment setup and implementation was done in simulation, where a satellite in low earth orbit with 4 different sensors were modelled. The purpose of the experiments were to evaluate the performance of the 3 different on-line attitude determination filters, and 2 different calibration filters, under different scenarios with a range of different sensor combinations. For this, metrics such as the MSE of the attitude estimate errors and calibration parameters along with computation time were compared.

The results show that we are able to estimate the calibration parameters for the gyroscope with a 99.06% accuracy, and that smoothing the estimates does not improve accuracy. The magnetometer calibration had a 99.98% accuracy with the UKF and a 96.84% accuracy with the EKF. Using the calibrated EKF values in the mission mode filters, we see that the SMEKF performs best in all scenarios with a similar computation time to that of the MMEKF. The regular MEKF is not useful for on-line implementation due to the similar performance to the MMEKF but with close to double the computation time.

Bibliography

[1] Malcolm David Shuster and S D_ Oh. "Three-axis attitude determination from vector observations". In: *Journal of guidance and Control* 4.1 (1981), pp. 70–77.

[2] Yanjun Xing, Xibin Cao, Shijie Zhang, et al. "Relative position and attitude estimation for satellite formation with coupled translational and rotational dynamics". In: *Acta Astronautica* 67.3-4 (2010), pp. 455–467.

[3] Joan Sola. "Quaternion kinematics for the error-state Kalman filter". In: *arXiv preprint arXiv:1711.02508* (2017).

[4] Yan-Bin Jia. "Quaternions". In: *Com S* 477 (2019), p. 577.

[5] R.L. Farrenkopf. "Analytic Steady-State Accuracy Solutions for Two Common Spacecraft Attitude Estimators". In: *Journal of Guidance and Control* 1.4 (1978), pp. 282–284. DOI: 10.2514/3.55779. eprint: https://arc.aiaa.org/doi/pdf/10.2514/3.55779. URL: https://arc.aiaa.org/doi/abs/10.2514/3.55779.

[6] F. Landis Markley and John L Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. eng. Vol. 33. Space Technology Library. New York, NY: Springer, 2014. ISBN: 9781493908011.

[7] P. Alken, E. Thébault, C. D. Beggan, et al. "International Geomagnetic Reference Field: the thirteenth generation". eng. In: *Earth, planets, and space* 73.1 (2021), pp. 49–25. ISSN: 1343-8832.

[8] Roberto Alonso and Malcolm D. Shuster. "Complete Linear Attitude-Independent Magnetometer Calibration". eng. In: *The Journal of the astronautical sciences* 50.4 (2002), pp. 477–490. ISSN: 0021-9142.

[9] Grace Wahba. "A least squares estimate of satellite attitude". In: *SIAM review* 7.3 (1965), pp. 409–409.

[10] Paul B Davenport. *A vector approach to the algebra of rotations with applications*. Vol. 4696. National Aeronautics and Space Administration, 1968.

[11] Jet Propulsion Laboratory. *JPL Planetary and Lunar Ephemerides*. https://ssd.jpl.nasa.gov/planets/eph_export.html. [Online; accessed December, 2022]. 2022.

[12] TC Van Flandern and KF Pulkkinen. "Low-precision formulae for planetary positions". In: *The Astrophysical Journal Supplement Series* 41 (1979), pp. 391–411.

[13] David A. Vallado. *Fundamentals of Astrodynamics and Applications*. eng. 4. ed. Microcosm Print, 2013. ISBN: 978-1881883180.

[14] Rafal Wiśniewski. *Lecture Notes on Modelling of a Spacecraft*. 2000.

[15] G. F. Franklin, J. D. Powell, and Micheal L Workman. *Digital control of dynamic systems.* eng. 3. ed. Addison-Wesley, 1998. ISBN: 0201331535.

[16] Mohinder S Grewal and Angus P Andrews. *Kalman filtering: theory and practice using MATLAB*. eng. 3rd ed. Wiley, 2008. ISBN: 9780470377802.

[17] Sai Jiang Fangjun Qin Lubin Chang and Feng Zha. *A Sequential Multiplicative Extended Kalman Filter for Attitude Estimation Using Vector Observations*. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5982416/. [Online; accessed November, 2022]. 2018.

[18] R Van der Merwe and E.A Wan. "The square-root unscented Kalman filter for state and parameter-estimation". eng. In: *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*. Vol. 6. IEEE, 2001, 3461–3464 vol.6. ISBN: 0780370414.

[19] G.H. Golub and C.F. Van Loan. *Matrix Computations*. 4th ed. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, 2013. ISBN: 9781421407944.

[20] H. E. RAUCH, F. TUNG, and C. T. STRIEBEL. "Maximum likelihood estimates of linear dynamic systems". In: *AIAA Journal* 3.8 (1965), pp. 1445–1450. DOI: 10.2514/3.3166. eprint: https://doi.org/10.2514/3.3166. URL: https://doi.org/10.2514/3.3166.

[21] John L. Crassidis, K. L. Lai, and Richard R. Harman. "Real-Time Attitude-Independent Three-Axis Magnetometer Calibration". In: *Journal of Guidance Control and Dynamics* 28 (2005), pp. 115–120.

[22] Space Inventor. *16U Satellite - Datasheet*. https://space-inventor.com/wp-content/uploads/2022/12/16U-space-inventor-catalouge.pdfhttps://gladiatortechnologies.com/g300d/. [Online; accessed December, 2022]. 2022.

[23] GomSpace. *NanoSense Fine Sun Sensor - Datasheet*. https://gomspace.com/UserFiles/Subsystems/datasheet/gs-ds-nanosense-fss-31.pdf. [Online; accessed December, 2022]. 2021.

[24] GomSpace. *NanoSense M315 - Datasheet*. https://gomspace.com/UserFiles/Subsystems/datasheet/gs-ds-nanosense-m315-11.pdf. [Online; accessed December, 2022]. 2017.

[25] Gladiator Technologies. *G300D Triaxial Gyroscope - Datasheet*. `https://gladiatortechnologies.com/g300d/`. [Online; accessed December, 2022]. 2022.