71st International Astronautical Congress (IAC) - The CyberSpace Edition, 12-14 October, 2020.

IAC–20–B2.7.11

IMPLEMENTATION AND VALIDATION OF MURRELL'S VERSION KALMAN FILTER FOR ATTITUDE ESTIMATION

Gaurav Sharma[a*], Tushar Goyal, Aditya Bhardwaj, Nikita Saxena, Jeet Yadav

[a] *Birla Institute of Technology and Science (BITS)-Pilani, India, f20170532@pilani.bits-pilani.ac.in*
[*] **Corresponding Author**

Abstract

Cubesats with imaging payloads face unique challenges in terms of stringent pointing accuracy and stability requirements. Team Anant is a student-run technical team working to build a 3U Cubesat. This paper discusses the implementation, validation and integration of an attitude estimation algorithm as part of the satellite's Attitude Determination System (ADS). The ADS hardware usually comprises sensors such as an IMU, magnetometer, and sun sensors. Validation methodology and architecture design, which aims to satisfy the allocated pointing budget, are also discussed. The paper introduces the motivation to choose Murrell's version Kalman Filter and a comparison with popular alternatives. This is followed by some prerequisites, after which, the paper describes the top level overview and testing framework developed for the kalman filter. This requires emulating the in-orbit environment and tracking the true state to establish the performance limit with a predefined performance metric. The verification procedure adopted by the team is discussed in detail. Apart from analysing the expected trend of the filter parameters over time, a quasi Monte Carlo (qMC) approach was also followed. Furthermore, the Cramer Rao Bound is used to establish a lower bound on the error covariance matrix. Lastly, an approach for fine sensor selection is provided based on emulating its integration with the ADS. The paper concludes by discussing the lessons learnt and the important stages in the development and testing of an attitude estimation algorithm.
Keywords: Attitude Estimation · Kalman Filter · Satellite Simulator · Monte Carlo · Pointing Metric

1. Introduction

Obtaining accurate and repeatable attitude information is an essential task for satellites with pointing requirements. This paper considers the implementation of an attitude estimation algorithm for a nanosatellite with an imaging payload. Attitude knowledge forms the basis for the pointing modes of the satellite [1], which are (a)Imaging (b) GS Tracking and (c) Sun Pointing.

The methodology followed to design an ADS for the cubesat has been elaborated in this paper. The need to develop and *validate* a robust system naturally arises. The hardware and software choices should satisfy any pointing accuracy requirements derived from the mission objectives.

Estimation algorithms provide a distinct advantage over determination techniques, and improve their performance by using past values to influence the current state estimate. The first section of the paper highlights the motivation behind selecting *Murrell's version MEKF* (Multiplicative Extended Kalman Filter) for the purposes of attitude estimation, in comparison with various other appproaches and variations. Most of these algorithms use quaternions as a standard attitude parametrization. The mathematical prerequisites follow this section and provide some context to the analysis and validation of the filter demonstrated in later sections of the paper.

Apart from implementing the filter itself, it is also crucial to emulate its expected working environment and the errors induced in the algorithm's input. Therefore, simulating the space environment and modelling the sensors accurately is essential to analyze its expected performance in orbit. The team has designed an indegeneous *Small Satellite Simulator* which provides the model with realistic inputs, while simultaneously tracking the true state of the satellite. The latter is essential to characterize the pointing accuracy and stability of the ADS. The pointing performance metric for the same is also detailed upon.

The accuracy and convergence of the filter depends on the initial estimate and the parameters used. For example, the initial error covariance matrix has a considerable effect on the filter. To optimize the performance of the filter given a particular set of hardware components, tuning of such parameters must be performed. The relevant parameters and the tuning approach for the initial error covariance matrix is elaborated upon, after which the results of the simulation are shown.

There are several ways to validate the operation and implementation of the kalman filter. A variant of the popular

Monte Carlo method was used to find a reliable estimate of the filter performance. The large number of runs take into account variations in the initial conditions and demonstrate the consistent behaviour of the filter. Additionally, the Cramér Rao bound is compared with the knowledge uncertainty to further validate the performace of the filter.

In the last section, we discuss the various challenges faced, and approaches that should be used to develop an attitude determination system. A process for selection of the high accuracy sensor is also described. The paper concludes by summarizing the results and elaborating on any future work.

2. Multiplicative Extended Kalman Filter

2.1 Overview

Kalman filter is an algorithm that provides estimates of some unknown variables, given the measurements observed over time [2]. In this case, it can be used to estimate the attitude of the satellite.

Quaternions are a popular choice for attitude parameterization because they have the minimum number of parameters that provides singularity free representation of SO(3) group. However, they must always follow the unit norm constraint, thus limiting their use in some cases. [3]

Assuming an unbiased estimate of the attitude quaternion, the additive representation of kalman filter leads to a violation of the normalization constraint. Furthermore, when the quaternion estimate is assumed to be biased, the additive representation results in an ill-conditioned error covariance matrix. Hence, a standard additive kalman filter is not suitable for attitude estimation using quaternions. This has been extensively discussed in [4].

Considering the lack of a robust solution to deal with these drawbacks, the Multiplicative Extended Kalman Filter (MEKF) was chosen to circumvent these problems.

Gyroscopes are crucial attitude sensors used in many satellites that require high attitude accuracy. While implementing MEKF, either a full gyro calibration can be performed or a simple bias estimation can be performed directly on the readings. [5]

Additionally, gain calculation in other MEKF filters require inverting a $3N \times 3N$ matrix where N is the number of measurement vectors available. Murrell's Version of Kalman Filter (now referred to as *Kalman Filter* in this paper) avoids this extensive computation by exploiting the principle of superposition, in which each vector observation is processed one at a time.

As detailed in the subsequent sections, each vector observation is processed separately, leading to inversion of a 3×3 matrix N times, instead of a $3N \times 3N$ matrix. This makes the selected approach computationally efficient.

2.2 Gyroscope Integration

One way to incorporate gyro errors in the filter is by modifying standard EKF equations to use gyro data as measurements. Theoretically, this should give better results but it usually performs poorly in practice. The alternative is to use gyro information directly in the dynamic model. This approach is called *Dynamic Model Replacement Mode*. [6].

The mathematical model for a three-axes rate integrating gyroscope has been described in [7] and is given by:

$$\omega(t) = \omega^{\mathbf{true}}(t) + \beta^{\mathbf{true}}(t) + \eta_{\mathbf{v}}(t) \quad [1]$$

$$\dot{\beta}^{\mathbf{true}}(t) = \eta_{\mathbf{u}}(t) \quad [2]$$

$$\mathbf{w}(t) = \left[\eta_{\mathbf{v}}^{\mathbf{T}}(t)\ \eta_{\mathbf{u}}^{\mathbf{T}}(t)\right]^T \quad [3]$$

where, $\omega^{\mathbf{true}}$ is the true angular velocity, ω is the measured angular velocity, and $\beta^{\mathbf{true}}$ is the true bias (or drift). $\eta_{\mathbf{u}}$ and $\eta_{\mathbf{v}}$ are independent zero-mean Gaussian white-noise processes with spectral density $\sigma_u^2\ \mathbf{I_3}$ and $\sigma_v^2\ \mathbf{I_3}$ respectively. $\mathbf{w}$ is a vector of Gaussian white noise processes.

A more robust calibration model of a general three-axes gyroscope would consist of 3 bias terms, and a 3×3 matrix consisting of 3 scale factors and 6 misalignments terms. These misalignment terms and scale factors have been ignored in this paper, due to computational considerations. However, to achieve more accurate results, these terms can also be incorporated. [7]

2.3 Murrell's Version Extended Kalman Filter

The basic idea of MEKF is to have two different attitude representations and then continuously switch between them. Since rotation vectors have no constraint on their norm, a three-component rotation vector $\delta\mathbf{v}$ will be used to represent *local* attitude error as part of state vector. Whereas quaternions will be used to represent the *global* attitude of satellite.

The true quaternion, in terms of quaternion error and estimated quaternion is given by:

$$\mathbf{q}^{\text{true}} = \delta\mathbf{q}(\delta\mathbf{v}) \otimes \hat{\mathbf{q}} \quad [4]$$

In the above equation,

$$\mathbf{A} \otimes \mathbf{B} = [\mathbf{A}\otimes]\mathbf{B}$$

where

$$[\mathbf{A}\otimes] = \begin{bmatrix} A_4 & A_3 & -A_2 & A_1 \\ -A_3 & A_4 & A_1 & A_2 \\ A_2 & -A_1 & A_4 & A_3 \\ -A_1 & -A_2 & -A_3 & A_4 \end{bmatrix}$$

Note that in Eq. [4] $\mathbf{q}^{\text{true}}$, $\delta\mathbf{q}(\delta\mathbf{v})$ and $\hat{\mathbf{q}}$ are all normalized. The state vector $\Delta\mathbf{x}$ will be

$$\Delta\mathbf{x}(t) = \begin{bmatrix} \delta\mathbf{v}(t) \\ \Delta\beta(t) \end{bmatrix} \quad [5]$$

where, $\Delta\beta(t) = \beta^{true} - \hat{\beta}$.

MEKF is based on linearizing non-linear systems. This causes the filter to be highly dependent on the initial value of the quaternion and the error covariance matrix. The initial quaternion in MEKF is provided by the QUEST algorithm. The effect of initial value of the error covariance matrix will be discussed in Section 4.3.

The Murrell's Version Kalman Filter has been implemented in four steps:

1. Propagation: Computes the apriori error covariance matrix using the attitude model.
2. Measurement Update: Revises the state vector and Kalman Gain using sensor measurements.
3. Reset: Transfers attitude information from local to global representation.
4. Quaternion Propagation: Propagate the global attitude quaternion to the next discrete timestep.

The following sections describe these four steps in detail.

2.3.1 Propagation

Time derivative of $\mathbf{q}^{\text{true}}$ in eq [4] is

$$\dot{\mathbf{q}}^{\text{true}} = \delta\dot{\mathbf{q}} \otimes \hat{\mathbf{q}} + \delta\mathbf{q} \otimes \dot{\hat{\mathbf{q}}} \quad [6]$$

True and estimated quaternions obey the following kinematic equations respectively:

$$\dot{\mathbf{q}}^{\text{true}} = \frac{1}{2}\begin{bmatrix}\omega^{\text{true}} \\ 0\end{bmatrix} \otimes \mathbf{q}^{\text{true}} \quad [7]$$

$$\dot{\hat{\mathbf{q}}} = \frac{1}{2}\begin{bmatrix}\hat{\omega} \\ 0\end{bmatrix} \otimes \hat{\mathbf{q}} \quad [8]$$

where ω^{true} and $\hat{\omega}$ are true and estimated angular velocities respectively. The above two equations imply that

$$\frac{1}{2}\begin{bmatrix}\omega^{\text{true}} \\ 0\end{bmatrix} \otimes \delta\mathbf{q} \otimes \hat{\mathbf{q}} = \delta\dot{\mathbf{q}} \otimes \hat{\mathbf{q}} + \frac{1}{2}\,\delta\mathbf{q} \otimes \begin{bmatrix}\hat{\omega} \\ 0\end{bmatrix} \otimes \hat{\mathbf{q}} \quad [9]$$

On post-multiplying $\hat{\mathbf{q}}^{-1}$ on both sides of the equation, and substituting $\omega^{\text{true}} = \hat{\omega} + \delta\omega$, the above equation can be approximated to:

$$\delta\dot{\mathbf{q}} = -\begin{bmatrix}\hat{\omega} \times \delta\mathbf{q}_{1:3} \\ 0\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\delta\omega \\ 0\end{bmatrix} \quad [10]$$

Using Taylor Series expansion, $\delta\mathbf{q}$ can be approximated as:

$$\delta\mathbf{q}(\delta\mathbf{v}) \approx \begin{bmatrix}\delta\mathbf{v}/2 \\ 1\end{bmatrix} = \mathbf{I}_{\mathbf{q}} + \frac{1}{2}\begin{bmatrix}\delta\mathbf{v} \\ 0\end{bmatrix} \quad [11]$$

After substituting Eq. [11] in the above equation, a simplified form is obtained i.e.

$$\delta\dot{\mathbf{v}} = -\hat{\omega} \times \delta\mathbf{v} + \delta\omega \quad [12]$$

The expectation of the above equation is

$$\delta\dot{\hat{\mathbf{v}}} = -\hat{\omega} \times \delta\hat{\mathbf{v}} \quad [13]$$

The state vector satisfies the following linearized dynamical equation

$$\Delta\dot{\mathbf{x}}(t) = \mathbf{F}(t)\,\Delta\mathbf{x}(t) + \mathbf{G}(t)\,\mathbf{w}(t) \quad [14]$$

By using Eq. [12], matrices $\mathbf{F}(t)$, $\mathbf{G}(t)$ can be obtained as

$$\mathbf{F}(t) = \begin{bmatrix}-[\hat{\omega}(t)\times] & -\mathbf{I}_3 \\ 0_{3\times3} & 0_{3\times3}\end{bmatrix} \quad [15]$$

$$\mathbf{G}(t) = \begin{bmatrix}-\mathbf{I}_3 & 0_{3\times3} \\ 0_{3\times3} & \mathbf{I}_3\end{bmatrix} \quad [16]$$

where

$$[\mathbf{V}\times] = \begin{bmatrix}0 & -V_3 & -V_2 \\ V_3 & 0 & -V_1 \\ -V_2 & V_1 & 0\end{bmatrix}$$

Spectral density $\mathbf{Q}(t)$ of $\mathbf{w}(t)$ is given by

$$\mathbf{Q}(t) = \begin{bmatrix}\sigma_v^2\mathbf{I}_3 & 0_{3\times3} \\ 0_{3\times3} & \sigma_u^2\mathbf{I}_3\end{bmatrix} \quad [17]$$

For discrete time Kalman Filter, the state transition matrix at the k^{th} time is given by

$$\phi_{\mathbf{k}} = \mathbf{I}_6 + \mathbf{F}_{\mathbf{k}}\Delta t \quad [18]$$

where $\mathbf{I}_6$ is 6×6 identity matrix and $\mathbf{F}_{\mathbf{k}}$ is obtained using [15].

The apriori error covariance matrix is then given by

$$\mathbf{P}_k^- = \phi_{\mathbf{k}}\mathbf{P}_{k-1}^+{\phi_{\mathbf{k}}}^T + \mathbf{G}_k\mathbf{Q}_k\mathbf{G}_k^T \quad [19]$$

where $\mathbf{G}_k$ and $\mathbf{Q}_k$ are obtained using [16] and [17] respectively.

2.3.2 Update

In this step, the state vector is updated. It is initialised to zero before the start of the step, as explained in Section 2.3.3. In the *Kalman Filter*, the update step occurs in batches depending on the number of sensor reading available at a given time. If N sensor readings are available at k^{th} time instant, and i denotes the i^{th} sensor reading, then:

$$\Delta\mathbf{x}_{k_i}^- = \begin{cases}\mathbf{0} & \text{if i = 1} \\ \Delta\mathbf{x}_{k_{i-1}}^+ & \text{if i} \neq 1\end{cases} \quad [20]$$

$$\Delta\mathbf{x}_k^+ = \Delta\mathbf{x}_{k_N}^+ \quad [21]$$

$$\mathbf{P}_{k_i}^- = \begin{cases} \mathbf{P}_k^- & \text{if i = 1} \\ \mathbf{P}_{k_{i-1}}^+ & \text{if i} \neq 1 \end{cases} \quad [22]$$

$$\mathbf{P}_k^+ = \mathbf{P}_{k_N}^+ \quad [23]$$

The measurement equation for *Kalman Filter* is given by

$$\mathbf{y}_k = \begin{bmatrix} \mathbf{A}(\mathbf{q}^{\text{true}})\mathbf{r}_1 \\ \mathbf{A}(\mathbf{q}^{\text{true}})\mathbf{r}_2 \\ \vdots \\ \mathbf{A}(\mathbf{q}^{\text{true}})\mathbf{r}_N \end{bmatrix} + \begin{bmatrix} \mathbf{n}_1 \\ \mathbf{n}_2 \\ \vdots \\ \mathbf{n}_N \end{bmatrix} = \mathbf{h}(\mathbf{x}_k^{\text{true}}) + \mathbf{n}_k \quad [24]$$

and

$$\mathbf{R} = \text{blkdiag}\begin{bmatrix} \mathbf{R}_1 & \mathbf{R}_2 & \dots & \mathbf{R}_N \end{bmatrix} \quad [25]$$

where blkdiag means block diagonal matrix, and $\mathbf{R}_i$ is the covariance of measurement noise $\mathbf{n}_i$.

The measurement errors in sensors are assumed to be isotropic, hence $\mathbf{R}_i = \mathbb{E}[\mathbf{n_i}\ \mathbf{n_i}^T] = \sigma_i^2\mathbf{I}_3$.

The true attitude matrix $\mathbf{A}(\mathbf{q}^{\text{true}})$ is related to the apriori attitude $\mathbf{A}(\hat{\mathbf{q}}^-)$ through

$$\mathbf{A}(\mathbf{q}^{\text{true}}) = \mathbf{A}(\delta\mathbf{q})\ \mathbf{A}(\hat{\mathbf{q}}^-) \quad [26]$$

The matrix $\mathbf{A}(\delta\mathbf{q})$ can be approximated as

$$\mathbf{A}(\delta\mathbf{q}) \approx \mathbf{I}_3 - [\delta\mathbf{v}\times] \quad [27]$$

For a single sensor, true and estimated body vectors are

$$\mathbf{b}_i^{\text{true}} = \mathbf{A}(\mathbf{q}^{\text{true}})\mathbf{r}_i \quad [28]$$

$$\hat{\mathbf{b}}_i^- = \mathbf{A}(\hat{\mathbf{q}}^-)\mathbf{r}_i \quad [29]$$

Hence, $\Delta\mathbf{b}_i$ is given by

$$\Delta\mathbf{b}_i = \mathbf{b}_i^{\text{true}} - \hat{\mathbf{b}}_i^- = -[\delta\mathbf{v}\times]\mathbf{A}(\hat{\mathbf{q}}^-)\mathbf{r}_i = [\hat{\mathbf{b}}_i^-\times]\delta\mathbf{v} \quad [30]$$

Using Taylor expansion it can be written as

$$\mathbf{h}_{k_i}(\mathbf{x}_{k_i}^{\text{true}}) \approx \mathbf{h}(\hat{\mathbf{x}}_{k_i}^-) + \mathbf{H}_{k_i}(\hat{\mathbf{x}}_{k_i}^-)\begin{bmatrix} \delta\hat{\mathbf{v}} \\ \Delta\hat{\beta} \end{bmatrix}_i \quad [31]$$

where $\mathbf{h}(\hat{\mathbf{x}}_{k_i}^-)$ is the estimated observation. From here on it follows that,

$$\begin{aligned} \mathbf{H}_{k_i}(\hat{\mathbf{x}}_{k_i}^-)\begin{bmatrix} \delta\hat{\mathbf{v}} \\ \Delta\hat{\beta} \end{bmatrix}_i &= \mathbf{h}_{k_i}(\mathbf{x}_{k_i}^{\text{true}}) - \mathbf{h}(\hat{\mathbf{x}}_{k_i}^-) \\ &= \mathbf{A}(\mathbf{q}^{\text{true}})\mathbf{r}_i - \mathbf{A}(\hat{\mathbf{q}}^-)\mathbf{r}_i \\ &= [\hat{\mathbf{b}}_i^-\times]\delta\mathbf{v} \end{aligned} \quad [32]$$

The measurement sensitivity matrix for k^{th} time and i^{th} sensor reading is given as

$$\mathbf{H}_{k_i}(\hat{\mathbf{x}}_k^-) = \begin{bmatrix}[\hat{\mathbf{b}}_i^-\times] & 0_{3\times3}\end{bmatrix} = \begin{bmatrix}[\mathbf{A}(\hat{\mathbf{q}}_k^-)\mathbf{r}_i\times] & 0_{3\times3}\end{bmatrix} \quad [33]$$

where, i varies from 1 to N.

The Kalman Gain is given by

$$\mathbf{K_{k_i}} = \mathbf{P}_{\mathbf{k_i}}^-\ \mathbf{H}_{\mathbf{k_i}}^{\mathbf{T}}\ (\mathbf{H_{k_i}}\mathbf{P}_{\mathbf{k_i}}^-\mathbf{H}_{\mathbf{k_i}}^{\mathbf{T}} + \mathbf{R_i})^{-\mathbf{1}} \quad [34]$$

Error covariance matrix and state vector are updated using the following equations

$$\mathbf{P}_{\mathbf{k_i}}^+ = [\mathbf{I_3} - \mathbf{K_{k_i}}\mathbf{H_{k_i}}]\ \mathbf{P}_{\mathbf{k_i}}^- \quad [35]$$

$$\Delta\mathbf{x}_{k_i}^+ = \Delta\mathbf{x}_{k_i}^- + \mathbf{K_{k_i}}\ [\mathbf{b_i} - \mathbf{A}(\hat{\mathbf{q}}_\mathbf{k}^-)\mathbf{r_i} - \mathbf{H_{k_i}}\Delta\mathbf{x}_{k_i}^-] \quad [36]$$

2.3.3 Reset

Kalman update step assigns a value to $\delta\hat{\mathbf{v}}^+$ and $\Delta\hat{\beta}^+$, but the global attitude still continues to be $\hat{\mathbf{q}}^-$ and $\hat{\beta}^-$. The reset step moves the attitude information gained from sensor measurements in the update step, from local attitude error to global variables. The equations that govern the reset step are:

$$\hat{\mathbf{q}}^+ = \delta\mathbf{q}(\mathbf{0})_3 \otimes \hat{\mathbf{q}}^+ = \delta\mathbf{q}(\delta\hat{\mathbf{v}}^+) \otimes \hat{\mathbf{q}}^- \quad [37]$$

$$\hat{\beta}^+ = \hat{\beta}^+ + \mathbf{0}_n = \hat{\beta}^- + \Delta\hat{\beta}^+ \quad [38]$$

A first order representation of the error quaternion in terms of error rotation vector is

$$\delta\mathbf{q}(\delta\hat{\mathbf{v}}^+) \approx \begin{bmatrix} \delta\mathbf{v}^+/2 \\ 1 \end{bmatrix} = \mathbf{I_q} + \frac{1}{2}\begin{bmatrix} \delta\mathbf{v}^+ \\ 0 \end{bmatrix} \quad [39]$$

This implies

$$\mathbf{q}^* \approx \left(\mathbf{I_q} + \frac{1}{2}\begin{bmatrix} \delta\mathbf{v}^+ \\ 0 \end{bmatrix}\right) \otimes \hat{\mathbf{q}}^- = \hat{\mathbf{q}}^- + \frac{1}{2}\delta\mathbf{v}^+ \otimes \hat{\mathbf{q}}^- \quad [40]$$

The above equation is used to get the updated global quaternion ($\hat{\mathbf{q}}^+$) by normalizing the approximation.

$$\hat{\mathbf{q}}^+ = \frac{\mathbf{q}^*}{\|\mathbf{q}^*\|} \quad [41]$$

Now that attitude information has been updated in global variable, the state vector can be set to zero.

$$\Delta\mathbf{x}_{k+1}^- = 0 \quad [42]$$

2.3.4 Quaternion Propagation

The discrete-time quaternion propagation is given by the following equation:

$$\hat{\mathbf{q}}_{k+1}^- = \exp[(\Delta\theta/2)\otimes]\hat{\mathbf{q}}_k^+ \approx \Theta(\omega_k^+)\hat{\mathbf{q}}_k^+ \quad [43]$$

where,

$$\Theta(\omega_k^+) = \begin{bmatrix} \cos\left(\frac{1}{2}\|\omega_\mathbf{k}^+\|\Delta t\right)\mathbf{I}_3 - [\hat{\psi}_k^+\times] & \hat{\psi}_k^+ \\ -\hat{\psi}_k^{+T} & \cos\left(\frac{1}{2}\|\omega_k^+\|\Delta t\right) \end{bmatrix} \quad [44]$$

and

$$\hat{\psi}_k^+ = \frac{\sin\left(\frac{1}{2}\left\|\boldsymbol{\omega}_{\mathbf{k}}^+\right\|\Delta t\right)}{\left\|\boldsymbol{\omega}_k^+\right\|} \quad [45]$$

All these steps, as shown in Figure 1, were implemented and tested in a framework developed by the team. The details of that framework are discussed in the next section.

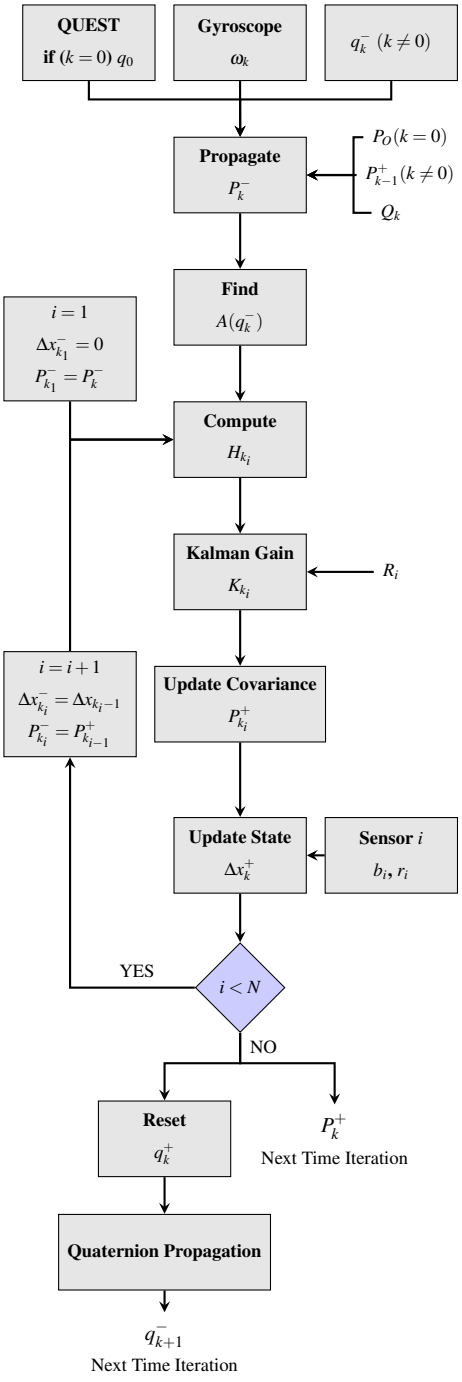

Fig. 1: Kalman Filter: Flow Chart

3. Testing Framework

3.1 Small Satellite Simulator

The team developed an in-house satellite simulator to accurately simulate the expected conditions in space [8]. Since the simulator was completely developed by the team, it was possible to add features that were of special interest.

Numerous such features have been used in this study to confirm the proper working of the filter. Some of them are as follows:

1. Satellite State Propagation (SSP) Model: An RK-4 based numerical integration method has been implemented to propagate the satellite's position, velocity and attitude.

2. Simplified Perturbations Model (SPM): An SGP based model has been implemented to simulate the various perturbations acting on the satellite while in orbit.

3. Environment Model: Various modules have been developed to simulate the environment conditions in the orbit:

   (a) *Sun Model*: Determines the position of Sun over time.

   (b) *Magnetic Model*: Determines the Earth's magnetic field based on the International Geomagnetic Reference Field (IGRF) Model.

   (c) *Albedo Model*: Determines the albedo caused by the sunlight reflected by Earth's surface.

   (d) *OLR Model*: Determines the Outgoing Longwave Radiation emitted by the Earth's surface.

4. Components Model: The different components onboard a satellite have been simulated as well, including various errors expected in their performance. The details of relevant sensor models have been discussed in Section 3.2.

   - Sun Sensors
   - Magnetometers
   - Gyroscopes
   - Accelerometers

The modular structure of the simulator also made it feasible to integrate the *Kalman Filter* with it easily.

3.2 Sensor Modelling

3.2.1 Sun Sensor

The true sun vector ($\mathbf{s}_t$) is obtained from the Small Satellite Simulator explained in 3.1. The angular error in

the sun sensor readings follow a normal distribution with $\mu = 0$ and $\sigma = \sigma_s$. This error is denoted by ϕ_s.

The cross product of a random unit vector $\mathbf{r}$ and $\mathbf{s}_t$ gives a vector $\mathbf{e}$. Using $\mathbf{e}$ and ϕ_s, a rotation matrix can be formed i.e.:

$$\mathbf{R}(\mathbf{e}, \phi_s) = \mathbf{I}_3 - \sin(\phi_s)[\mathbf{e}\times] + (1 - \cos(\phi_s))[\mathbf{e}\times]^2 \quad [46]$$

where, $\mathbf{e}$ is the euler axis of $\mathbf{R}$ and ϕ_s is the euler angle.

The final sun sensor reading ($\mathbf{s}_m$) can be determined by rotating the true vector using $\mathbf{R}$, as shown in Figure 2.

$$\mathbf{s}_m = \mathbf{R}(\mathbf{e}, \phi_s)\ \mathbf{s}_t \quad [47]$$

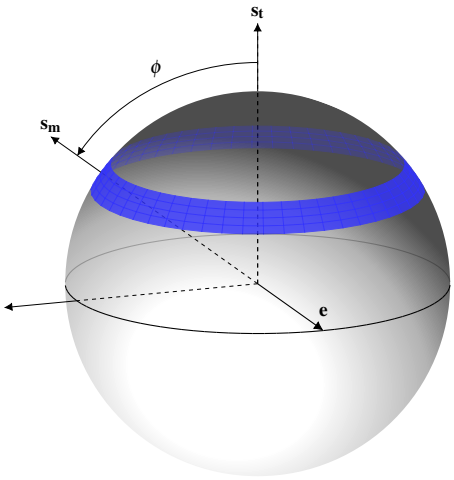

Fig. 2: Erroneous vector representation

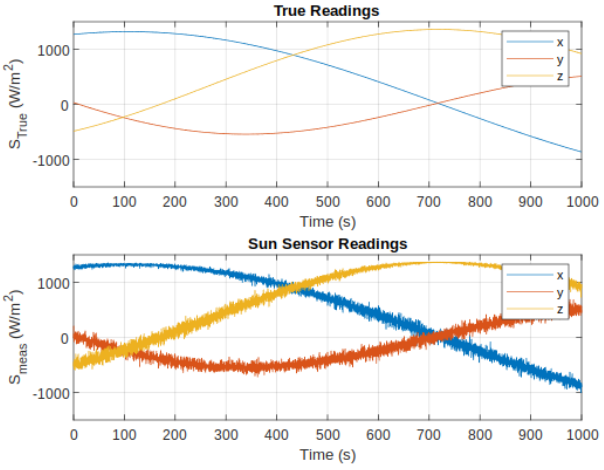

Fig. 3: Sun Sensor Model

Given the error angle ϕ_s drawn from a probability distribution, the blue sphere segment shows the equiprobable locations of the erroneous vector. The final readings of the sun sensor are shown in Figure 3.

3.2.2 *Magnetometer*

MATLAB Sensor Fusion and Tracking Toolbox was used to model the magnetometer. Functions *magparams* and *imuSensor* were utilized for the same.

Parameters such as *Rate Noise Density* and *Measurement Range* were selected based on the chosen magnetometer *PNI RM3100* [9]. *Random Walk* and *Bias Instability* were not taken into consideration. Temperature bias and scale factor were assumed to be zero.

The sensor was simulated with the parameters specified in Table 1.

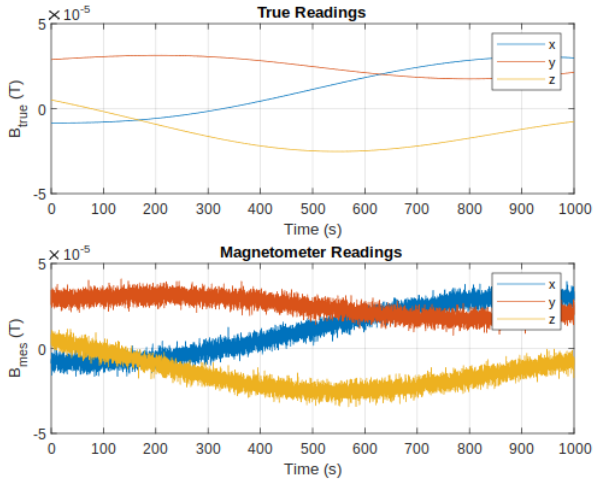

Fig. 4: Magnetometer Model

Table 1: Magnetometer Parameters

Parameter	Value
Measurement Range	800 μT
Rate Noise Density	$4 \times 10^{-6}\ \mu T\ Hz^{-1/2}$

The sensor readings were as shown in Figure 4.

3.2.3 *Gyroscope*

MATLAB Sensor Fusion and Tracking Toolbox was also used to model the gyroscope. Functions *gyroparam* and *imuSensor* were utilized for the same.

As with the magnetometer, temperature effects on readings were ignored for simulating the gyroscope as well. The parameters of the model are based on the gyroscope *ADIS 16475*, as shown in Table 2.

Table 2: Gyroscope Parameters

Parameter	Value
Random Walk	4.360×10^{-5} $rad\ s^{-1}Hz^{-1/2}$
Bias Instability	9.696×10^{-6} $rad\ s^{-1}$
Rate Noise Density	5.230×10^{-5} $rad\ s^{-1}Hz^{-1/2}$

The sensor readings were as shown in Figure 5.

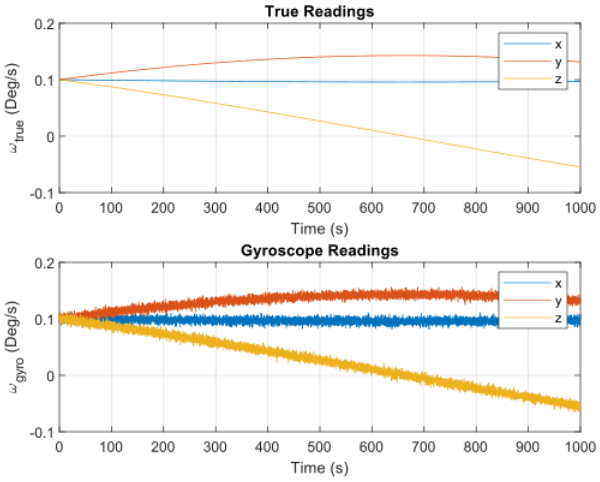

Fig. 5: Gyroscope Model

3.3 Integration

The *Kalman Filter* is implemented by using various inputs from the Small Satellite Simulator (Section 3.1) and Sensor Models (Section 3.2). These connections are shown in Figure 6.

Since the performance of the *Kalman Filter* is highly dependent on the choice of initial estimate, the QUEST algorithm is utilized for it. [10]

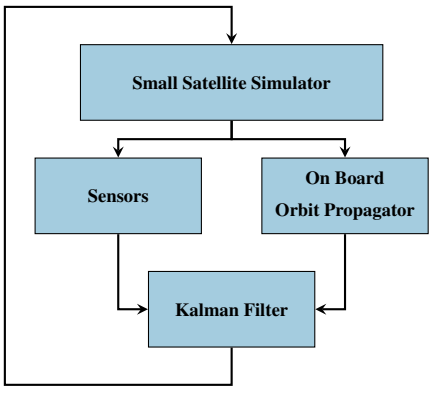

Fig. 6: Information flow between blocks

The inputs to the Sensor Model Block provided by the Simulator are given below. All of them are represented in the body frame.

- Sun Intensity Vector: As an input to sun sensor.
- Magnetic Field Vector: As an input to magnetometer.
- Angular velocity of the satellite: As an input to gyroscope.

The inputs to the *Kalman Filter* provided by the Simulator are:

- Sun Intensity Vector (ECI): $\mathbf{r}_1$
- Magnetic Field Vector (ECI): $\mathbf{r}_2$

The Sensor Model provides the following inputs to the *Kalman Filter*:

- Sun Sensor reading: $\mathbf{b}_1$
- Magnetometer reading: $\mathbf{b}_2$
- Gyroscope reading: ω_k

The *Kalman Filter* obtains the quaternion for the first iteration from the QUEST algorithm. The other inputs to the *Kalman Filter* are:

- Tuned initial Error Covariance Matrix (P_0) (Section 4.3).
- Measurement Noise Covariance Matrix (R), based on sensor error (σ_i).
- Process Noise Covariance Matrix (Q), based on σ_u and σ_v of the gyroscope.

The output of the *Kalman Filter* is eventually comapred against the true ECI-to-Body quaternion received from the Small Satellite Simulator. The whole process is described as a flowchart in Figure 1.

4. Implementation

4.1 Pointing Metric

The attitude of a spacecraft must satisfy pointing and stability requirements which are derived from the science objectives and other pointing modes. Analysis was focused on imaging, since that requires tracking the orientation (instead of a vector) and also has a stricter pointing criterion.

Attitude estimation is imperfect and contributes it's share of error; this pointing error source (PES) is allocated a certain error budget. This requires defining an appropriate metric and testing the estimation against it. The ESA pointing standards[11] are used to establish a pointing metric for pointing *knowledge* error.

Figure 7 shows the relevant time intervals over which we measure absolute and relative, pointing and stability knowledge.

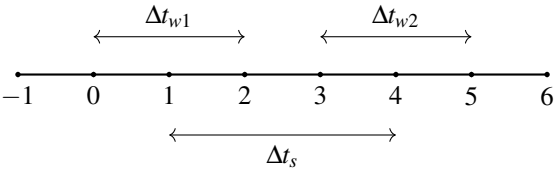

Fig. 7: Time intervals

- **Window time Δt_w:** Time interval over which pointing errors can be observed and averaged. Relative pointing error is defined for any time instant, relative to the mean pointing error in one Δt_w.
- **Stability time Δt_s:** Mean of the time windows are separated by a time difference of Δt_s, as shown in Figure 7. This is used to observe the drift and repeatability of the pointing errors between different time windows.

The absolute knowledge error $e_k(t)$ is the difference between the true and estimated parameters. It is worth noting at this point that the error parametrization is arbitrary to an extent and would depend on the usage. As an example, for sun pointing the half angle around the sun vector would suffice. However, since imaging is more stringent, the parameter selected is the *error* euler angle corresponding to the *error* quaternion.

The *Mean Knowledge Error* (MKE) is the mean value of the absolute knowledge error $e_k(t)$ over the specified time interval Δt_w.

$$\text{MKE} = \bar{e}_k(t, \Delta t_w) \quad [48]$$

Assuming the filter has converged to a stable value, the pointing accuracy can further be defined as a mean of the MKE itself over all time windows.

The *Knowledge Drift Error* (KDE) is the difference between MKE's taken over two time intervals, separated by the stability time Δt_s, within a single observation period. Similar to accuracy, the pointing stability (after convergence) can be defined as a mean of the successive KDEs.

$$\text{KDE} = \bar{e}_k(t, \Delta t_1) - \bar{e}_k(t + \Delta t_s, \Delta t_2) \quad [49]$$

Hence, the error can be characterized by plotting MKE/KDE versus time or taking a mean value of these after convergence. The above definitions are implicit when we talk about accuracy and stability in the later sections.

4.2 Effect of Error Covariance Matrix

It was observed that the performance of the *Kalman Filter* heavily depended on the initial value of the error covariance matrix (P_0), given by:

$$P_0 = \begin{bmatrix} p_q\, I_3 & 0_{3\times3} \\ 0_{3\times3} & p_b\, I_3 \end{bmatrix} \quad [50]$$

where I_3 is 3×3 identity matrix, $0_{3\times3}$ is a 3×3 zero-matrix, p_q is the Error Covariance Matrix for error vector ($\delta\mathbf{v}$) and p_b is the Error Covariance Matrix for bias ($\Delta\beta$) as described in Eq. [5]

The importance of the P_0 has been noted in [12, 13, 14].

Due to the highly non-linear nature of attitude estimation, the filter diverges if the diagonal elements of P_0 are very small. The *Kalman Filter* relies more heavily on the prediction if the value of P_0 is small. However, the inaccuracy of the prediction causes the filter to diverge. In contrast, if P_0 is increased to a high value. it results in a singular value for updated matrix P_k^+. Hence proper tuning of P_0 is crucial for optimal filter performance.

4.3 Tuning

Mean Knowledge Error and *Knowledge Drift Error*, as defined in Section 4.1 are used to describe the accuracy and stability of the *Kalman Filter* post-convergence. The filter is assumed to be converged when successive KDEs falls below a certain threshold. The methodology involved in evaluating the performance of the *Kalman Filter* is detailed below.

The absolute knowledge error, $e_k(t)$ is measured over a single observation period t_n. The MKE is computed over consecutive minute-long time windows. As defined in Section 4.1, this implies:

$$\Delta t_w = \Delta t_s = 60 \text{ s} \quad [51]$$

In the current implementation, the *Kalman Filter* is assumed to have converged when the KDE falls below a threshold of 0.1 at least 10 times. The time of convergence is denoted as t_c.

Post convergence, the mean and standard deviation of the MKEs is computed:

$$\mu = \frac{\sum_{i=1}^{N} MKE(t_i)}{N} \quad [52]$$

$$\sigma^2 = \frac{\sum_{i=1}^{N} MKE^2(t_i)}{N} - \mu^2 \quad [53]$$

where N is the number of minute-long intervals post-convergence.

Eq. 52 can be correlated with the accuracy of the *Kalman Filter*. The lower the mean of MKEs, the higher the accuracy. Similarly, Eq. 53 correlates with the stability of the *Kalman Filter*. The variance measures the deviation in the $e_k(t)$ from the mean value. Lesser the deviation, greater the stability of the *Kalman Filter* post-convergence. Based on the mission requirement the value of required

accuracy, required stability and required convergence time is obtained. The initial value of P_0 as described in previous subsection, depends on p_q and p_b which are defined as:

$$p_q = 10^{-a}\mathbf{I_3} \quad [54]$$

$$p_b = 10^{-b}\mathbf{I_3} \quad [55]$$

The initial error covariance matrix P_0 was changed by varying the dependent parameters a and b within certain intervals. The accuracy, stability and convergence time was computed for each case. The optimal P_0 is obtained when these values are within the defined requirements.

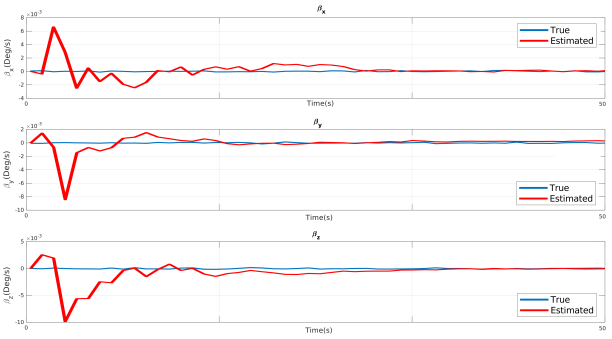

Fig. 8: Gyroscope Bias ($^\circ/s$) vs Time (s)

4.4 Implementation Results

The simulation for the *Kalman Filter* were performed using the Small Satellite Simulator (Section 3.1), and the chosen sensors (Section 3.2). The simulation was run for 2500 seconds with a time-step of 0.1 seconds (10 Hz), which is compatible with the operation rates of the sensors.

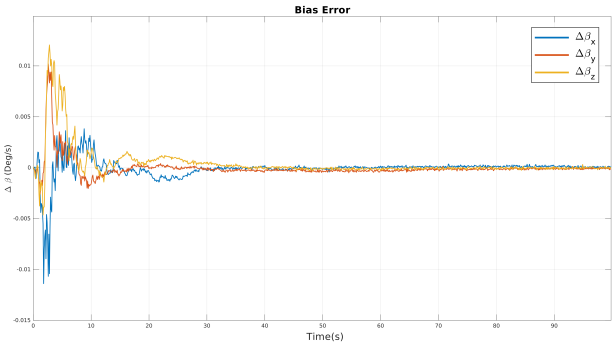

Fig. 9: Gyroscope Bias Error ($^\circ/s$) vs Time (s)

4.4.1 Error Definition

Consider the estimated and true body frames. Both of these are parametrized with respect to the reference frame using the attitude quaternions $\hat{\mathbf{q}}$ and $\mathbf{q}^{true}$ respectively.

To characterize and visualize the performance of the kalman filter, the euler angle of the *error* quaternion is calculated. The error quaternion $\delta\mathbf{q}$ represents the orientation of the true body frame with respect to the estimated body frame, as given in Equation 4.

$$\theta_e = 2\ cos^{-1}([\ \delta\mathbf{q}\ ]_1) \quad [56]$$

Where $[q]_1$ denotes the scalar part of the quaternion.

The error euler angle θ_e is used in the following sections to verify the convergence values of the filter. Alternatively, euler angles (Roll Pitch Yaw) may also be used to reflect this error about different axes.

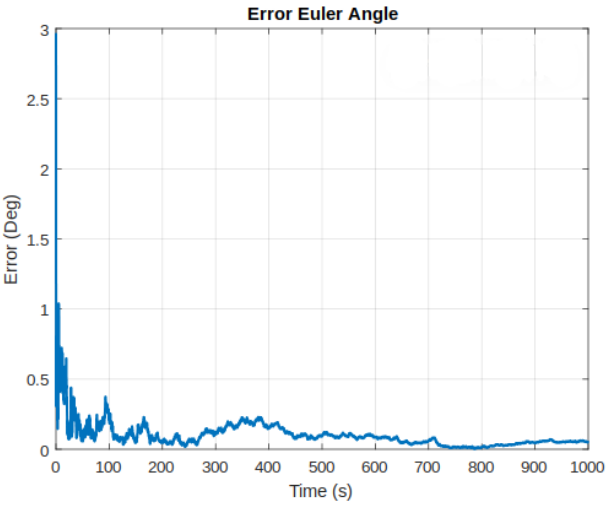

Fig. 10: Euler Error Angle ($^\circ$) vs Time (s)

4.4.2 Results of Filter Run

The plots for bias, $\Delta bias$, θ_e and the quaternion components are given in Figure 8, 9, 10, 11 respectively. It can be seen that the *Kalman Filter* converged well within the simulation time, and was successfully able to track the attitude of the satellite.

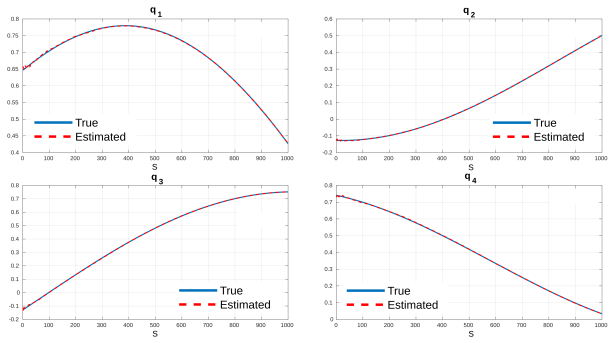

Fig. 11: Quaternion vs Time (*s*)

4.4.3 Results of Tuning

Figure 12 shows the post-convergence error in the *Kalman Filter* for different initial Error Covariance Matrices (P_0). It is abundantly clear that the performace of the filter improves when using a tuned P_0 in comparison to an untuned one. It can also be seen that the performace of the filter degraded beyond control while using an extreme P_0.

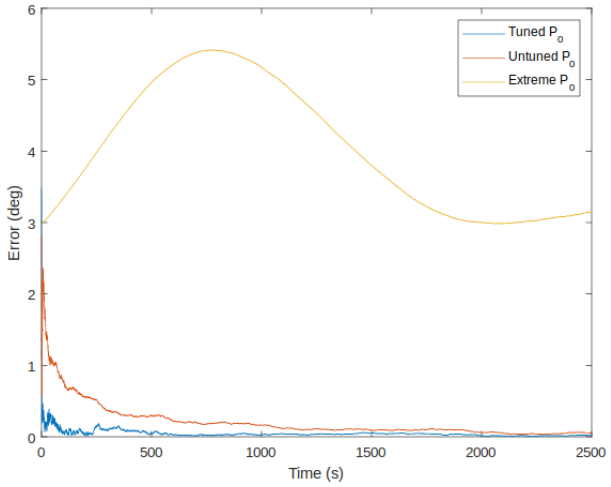

Fig. 12: Performance based on Initial Error Covariance Matrix Tuning (*s*)

5. Validation

5.1 Inspection

A fundamental procedure to validate the performance of a Kalman Filter involves observing the behavior of the trace of the error covariance matrix ($\mathbf{P_k^+}$) and Frobenius Norm of Kalman Gain ($\mathbf{K_k}$). [15]

A decrease in $\mathbf{P_k^+}$ at each timestep implies a corresponding decrease in the absolute knowledge error. It can be visualised by plotting the variation in the trace of $\mathbf{P_k^+}$. This was observed for an orbit. The variation for first few timesteps is shown in Figure 13.

With each successive time instant, an optimal Kalman Filter should learn to rely more on the estimate (via the process) as compared to the measurement. Therefore, Eq. 34 implies that the (Frobenius Norm of) Kalman Gain should decrease over time. Figure 14 shows the behaviour of Frobenius Norm of the Kalman Gain for a few timesteps. These results give credence to the implemented *Kalman Filter*.

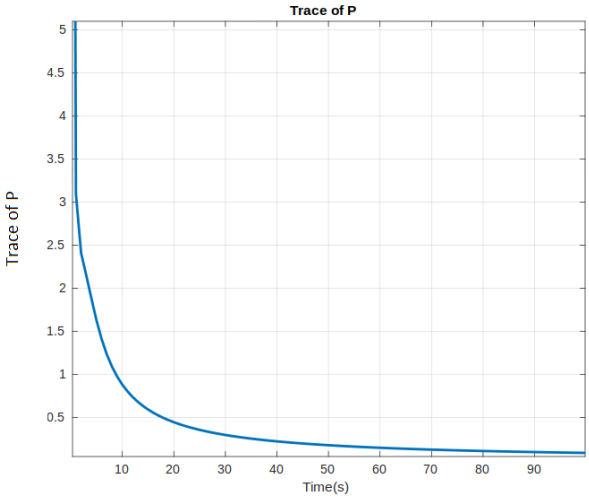

Fig. 13: Trace of Error Covariance Matrix vs Time (*s*)

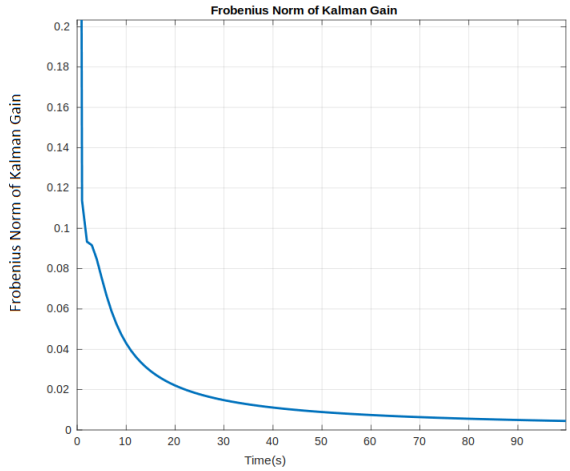

Fig. 14: Frobenius Norm of Kalman Gain vs Time (*s*)

5.2 Quasi-Monte Carlo Simulations

Monte Carlo Simulation (MCS) is a method to study the probability of different outcomes in a process which relies heavily on a large input space of variables. It is a popular technique used to understand the impact of risk and uncertainty, especially in prediction and estimation models.

A typical MCS involves the selection of input variables from a pseudo-random set of possible values, for each run of the simulation. The large number of initial conditions drawn from the sample space help paint a semi-exhaustive picture of the performance of the test model.

Due to the need for running multiple iterations to avoid inaccurate results, running a MCS can be computationally expensive and time-consuming. Since the input variables are supposed to be representative of the function space, the pseudo-random generation leads to clumping and convergence errors. To circumvent this problem, the team decided to approach the initial sampling differently, following a Quasi-Monte Carlo method.

Quasi-Monte Carlo simulation is identical to the traditional version, but uses quasi-random sequences instead of pseudo-random numbers. Such sequences have a low-discrepancy, which reflects on the equidistributed nature of the samples. This offers a better performance against traditional pseudo random sequences for all four probabalistic moments (Mean, Variance, Skewness, Kurtosis). Thus a smaller and more uniform set of initial conditions can be used to analyze the performance of the *Kalman Filter*.

Along with requiring the initial conditions for each run, we also need to track the true state of the satellite to provide a base to measure the performance of the filter. This true state is deterministic and independent of the filter itself. The next subsection describes the method to generate the relevant parameters and the corresponding data sets.

5.2.1 Data Set Creation

Each Data set comprises the following parameters, which need to be initialized and tracked through the attitude estimation process.

1. **True Quaternion** of the Body wrt ECI frame
2. **Angular Velocity** of the Body wrt ECI Frame, represented in the Body frame
3. **Measurement Vectors** for the purpose of attitude determination, both in the Body and ECI Frame. In this case, its the sun vector and magnetic field.

A program was created to choose the initial parameters from a quasi-random pool, and propagate them to provide the true state variables for all time instances. The following approach was undertaken for each parameter. Keep in mind that initial values are propagated through the appropriate rotational dynamics for the body.

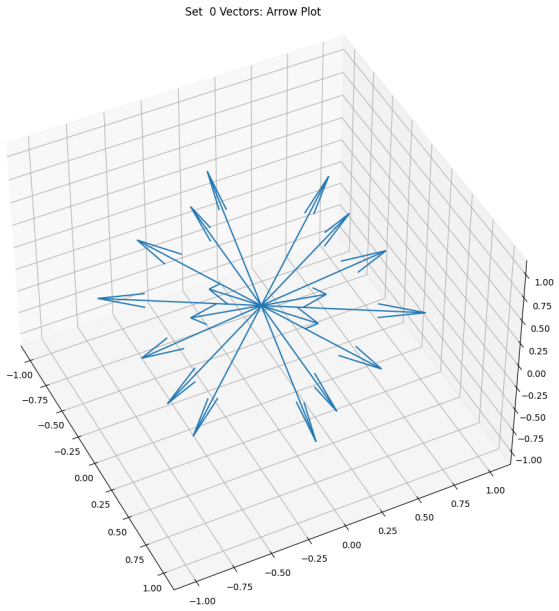

Fig. 15: S_0 Vector Distribution

1. **Angular Velocity**: 18 uniformly-distributed vectors were taken on the 3D sphere. This set was named S_0. All the 18 vectors in S_0 were then given a constant magnitude and used as the initial angular velocity vector.
2. **True Quaternion**: An arbitrary but constant initial orientation is selected for all iterations.
3. **Sensor Values (Body)**: Similar to S_0, identically spaced sets are created for the measurement sensors. S_n (where n is the index of the sensor) is different from S_0 by a constant, randomized rotation. Each such set was then used as different possible initial directions for different sensor readings.
4. **Sensor Values (Reference)**: The initial sensor values in the body frame, and true quaternion were used to find the corresponding vectors in the reference frame.

Figure 15 shows the uniformly-spaced distribution of the 18 vectors on a 3D sphere.

The assumptions taken for the propagation of these variables are given below.

- The body is perfectly spherical, with a uniform density.
- The reference frame remains unchanged for the complete duration of a simulation.
- The error in reference frame vectors used by the filter is negligible.

18^3 datasets were created for simulations. For each of them, the variables were propagated for a total of 2500 time steps.

5.2.2 Simulation Result

The dataset produced in the previous section was then used to run the *Kalman Filter* for multiple iterations. The average error in the filter was then noted with respect to time, and is plotted in Figure 16. This error decreases over time, thus verifying the performance of the filter over multiple quasi-random initial conditions.

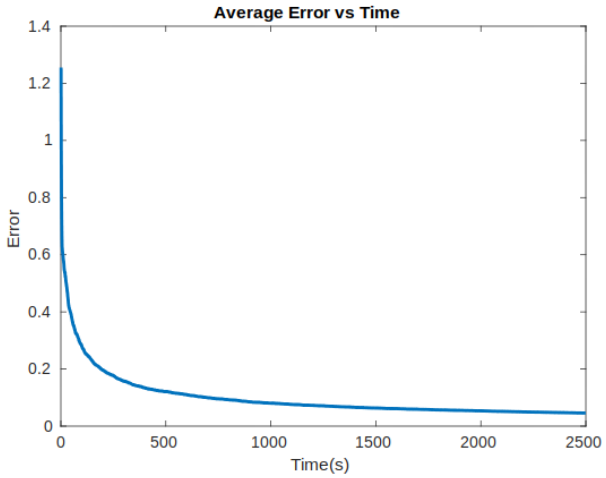

Fig. 16: Average Error in Kalman Filter (°) vs Time (s)

5.3 Cramér Rao Bound

5.3.1 Theory

Cramér Rao Bound of an unbiased estimator is the lower bound on its variance. The Cramér Rao Lower Bound (CRLB), denoted by $\mathbf{C}_k^+$, is equal to the inverse of Fisher Information Matrix.

Equations to compute and propagate CRLB for the *Kalman Filter* are discussed in this section. Another technique to analyse the operation of a filter, is to ensure that it always satisfy the following inequality:

$$\mathbf{\Pi}_k^+ \geq \mathbf{C}_k^+ \quad [57]$$

where, $\mathbf{\Pi}_k^+$ is the *Mean Square Error Matrix*, which is computed using Equation [58].

$$\mathbf{\Pi}_k^+ \approx \frac{1}{M}\sum_{i=1}^{M} \alpha_k^+(i)(\alpha_k^+)^T(i) \quad [58]$$

where $\alpha_k^+(i) = \mathbf{x_k^{true}}(i) - \hat{\mathbf{x}}_\mathbf{k}^+(i)$ is the estimate error at the i^{th} Monte Carlo Simulation.

Given that both matrices are symmetric, Eq. [57] implies that the eigenvalues of $\mathbf{\Pi}_k^+ - \mathbf{C}_k^+$ should always be positive. This suggests the following equation, although the reverse condition is not necessarily true.

$$\text{tr}(\mathbf{\Pi}_k^+) \geq \text{tr}(\mathbf{C}_k^+) \quad [59]$$

The following equations can be used to recursively compute CRLB for all time instances. [16]

$$\mathbf{A} = \mathbf{H}_k\mathbf{C}_k^-\mathbf{H}_k^T + \mathbf{R}_k \quad [60]$$

$$\mathbf{C}_k^+ = \mathbf{C}_k^- \mathbf{C}_k^- \mathbf{H}_k^T \mathbf{A}^{-1} \mathbf{H}_k \mathbf{C}_k^- \quad [61]$$

$$\mathbf{C}_{k+1}^- = \mathbf{F}_k\mathbf{C}_k^+\mathbf{F}_k^T + \mathbf{G_k}\mathbf{Q}_k\mathbf{G}_k^T \quad [62]$$

In above equation $\mathbf{C}_{k+1}^-$ is intialized as $\mathbf{C}_0^- = \mathbf{P}_0^-$.

5.3.2 Simulation Result

In Figure 17 we can see that all the eigenvalues of difference matrix i.e. $\mathbf{\Pi}_k^+ - \mathbf{C}_k^+$ are positive and decreasing over all times. This implies that our filter estimate improve with time step and error co-variance matrix ultimately approaches the Cramér-Rao Lower Bound.

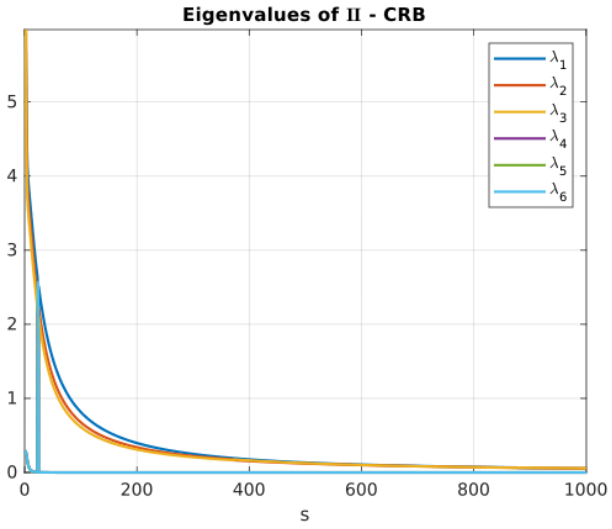

Fig. 17: Eigen Values of $\mathbf{\Pi}_k^+ - \mathbf{C}_k^+$

As expected, in Figure 18 we can see that the inequality given in Eq. [59] is also satisfied by our filter.

This result validates the performance of the designed *Kalman Filter*.

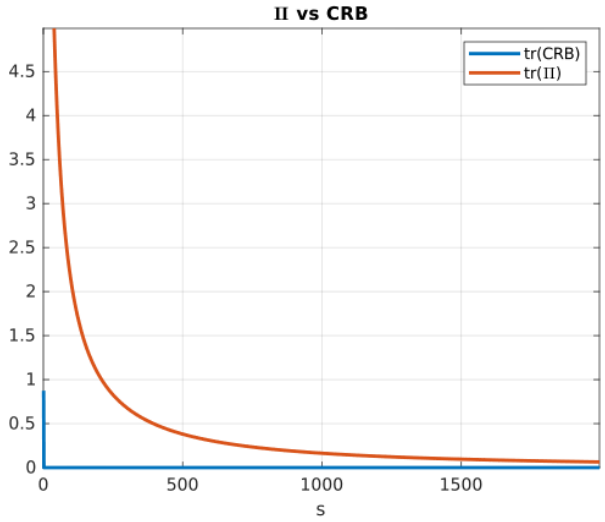

Fig. 18: Trace $\mathbf{\Pi}_k^+$ and $\mathbf{C}_k^+$

6. Results and Discussions

In Section 4.4, the results for the designed *Kalman Filter* were plotted. The error euler angle was observed to measure the Filter's performance. The gyroscopic bias and the error associated with it was also plotted. From the plots it is clear that the *Kalman Filter* is converging to an accurate estimate.

As shown in Section 4.2, the filter's accuracy, stability and convergence time were affected by the choice of $\mathbf{P_0}$. Hence, the $\mathbf{P_0}$ was tuned to obtain the optimal value, which outperforms the untuned and extreme case values of $\mathbf{P_0}$ by considerable margins.

To validate the *Kalman Filter*, the trace of P and the norm of Kalman Gain was plotted with time. Both the quantities were observed to be monotonously decreasing. By performing quasi-Monte Carlo simulations in Section 5.2, the filter was tested for a large number of test cases in which measurement vector and angular velocity spanned over the entire 3D sphere. By doing so, a robust testing of the filter was done and long term behaviour of the filter was assessed. It was found that the Filter performed as expected.

Finally in Section 5.3, we discuss the implementation and results of Cramér-Rao Bound Method. We find that the filter was behaving as expected and was satisfying the inequalities described in Eq. [57] and Eq. [59]. The fact that the eigenvalues of difference between MSME and CRB were decreasing with time shows that the filter is converging to the optimal estimate.

The numerous implementation and validation tests performed, confirm the stability, accuracy and robustness of the *Kalman Filter*. Furthermore, the study can be used to determine the required specifications for various sensors, as detailed in Section 6.1.

6.1 Selection of Fine Sun Sensor

Hardware constraints heavily affect the design strategy for nanosatellites, due to the stringent size and power constraints. For high accuracy ADCS systems, a fine sun sensor or star tracker must be selected which satisfies both pointing and technical budgets of the satellite. These components are usually quite large and consume a lot of energy compared to other components and sensors.

A selection methodology is shown in Figure 19, which would use the already developed attitude estimation system to check whether the sensor is compatible with the design requirements. This assumes the choice of other hardware and estimation algorithm, and thus translates the sensor performance to the system operation. The performance metric for attitude estimation, as described in Section 4.1, can be used for direct comparison with the allocated pointing budget.

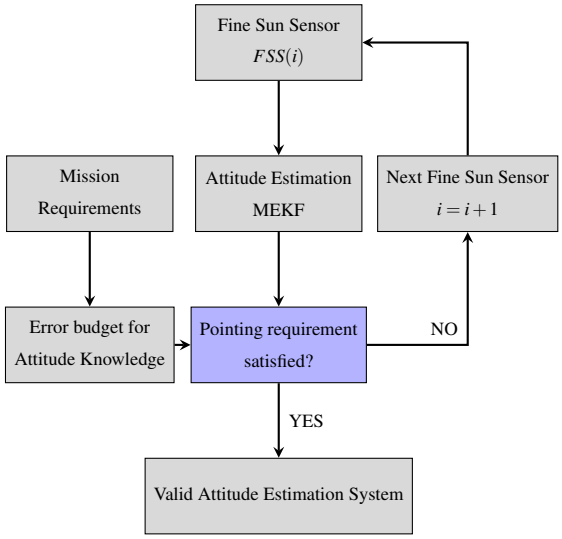

Fig. 19: Approach to select the Fine Sensor

Keep in mind that the MEKF is implictly tuned to give the best performance for a particular sensor. Furthermore, qMC simulations can be run to obtain a reliable performance limit for the hardware.

The block diagram shown in Figure 19 is iterative and evaluates all the fine sun sensors under consideration. For multiple sensors which satisfy the requirements, the decision can be based on the form factor and power consumed by each.

7. Conclusion

This paper puts forth an implementation of Murrell's version of Extended Kalman Filter. A preliminary literature review was conducted and various implementations were compared to choose the best option based on the scientific objective. The chosen implementation was derived and theoretically verified. A dynamic satellite simulator, along with realistic sensor models were developed to perform various tests. The filter was then implemented, and parameters describing the convergence and tuning of the filter were examined. A metric to determine the accuracy and stability of the filter was also devised. The performance of the filter was well within the expected range.

To validate the working of the filter for all possible cases, a quasi Monte-Carlo simulation was also performed. Cramér Rao Bound Method was then executed to establish a lower bound on the error covariance matrix. The behaviour of the norm of error covariance matrix further verified the convergence of the filter. Finally, the data collected from the test runs was used to determine the required specifications of a potential fine sun sensor. The work presented in this study enabled the team to find the optimal sensor based on the developed Kalman Filter.

The future work includes integrating the *Kalman Filter* with control inputs and to perform a hardware in loop testing so as to more critically analyse the performance of the filter. This will be followed by integrating the *Kalman Filter* with the Modes of Operation of Satellite.

Acknowledgements

The authors would like to express there sincere gratitude towards the following:

1. Team Anant, the Student Satellite Team of BITS Pilani, for providing the motivation and resources for the work done in this paper.
2. Dr. Kaushar Vaidya and Dr. Meetha V. Shenoy for their invaluable guidance throughout the research.
3. Vishnu Katkoori, Jivat Neet Kaur, George Savio and Amay Sareen for their constant support and technical expertise in persuing research for this paper.
4. The administration of BITS Pilani and Indian Space Research Organization (ISRO) for giving us an opportunity to work on building a satellite.

References

[1] R. Jain et al. Modes of operations for a 3u cubesat with hyperspectral imaging payload. In *2018 International Astronautical Congress*, 2018.

[2] Kalman R.E. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering, ASME*, 1960.

[3] Landis Markley. Attitude error representations for kalman filtering. *Journal of GCD*, 26:311–317, 2003.

[4] F. Landis Markley and John L. Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, New York, 2014.

[5] J.W. Murrell. Precision attitude determination for multimission spacecraft. *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, page 70–87, 1978.

[6] John Crassidis and Landis Markley. Three-axis attitude estimation using rate-integrating gyroscopes. *Journal of GCD*, 39:1–14, 2016.

[7] Landis Markley and R. Reynolds. Analytic steady-state accuracy of a spacecraft attitude estimator. *Journal of GCD*, 23, 2000.

[8] T. Goyal and K. Aggarwal. Simulator for functional verification and validation of a nanosatellite. In *2019 IEEE Aerospace Conference*, pages 1–8, 2019.

[9] Leonardo H. Regoli. Investigation of a low-cost magneto-inductive magnetometer for space science applications. *Geosci. Instrum. Method. Data Syst.*, (7):129–142, 2018.

[10] M.D. Shuster. The quest for better attitudes. *J of Astronaut Sci.*, pages 657–683, 2006.

[11] ESSB-HB-E-003 Working Group. *ESA Pointing Error Engineering Handbook*. ESA, Europe, 2011.

[12] Maybeck P S. *Stochastic models, estimation, and control. Volume 1*. Academic Press, New York, 1979.

[13] M.R. Ananthasayanam. A heuristic reference recursive recipe for adaptively tuning the Kalman filter statistics. *Sādhanā*, (41):1473–1490, 2016.

[14] M.R. et al. Mohan M, Ananthasayanam. Introduction to the kalman filter and tuning its statistics for near optimal estimates and cramér rao bound. *arXiv*, (1503), 2015.

[15] X. Hu. Generalized Iterated Kalman Filter and its Performance Evaluation. *IEEE Transactions on Signal Processing*, 63(12):3204–3217, 2015.

[16] Jindich Havlík and Ondej Straka. Performance evaluation of iterated extended Kalman filter with variable step-length. *J. Phys.*, (659), 2015.