Assignment V: Advanced Robot Navigation (RBE 595)

Pradnya Sushil Shinde

April 30, 2024

1 INS/GNSS Intergration

The objective of the project is to implement an INS/GNSS integration scheme using a nonlinear Kalman Filter (UKF). The State Model will remain the same as used in the previous assignment of the Non-Linear Kalman Filter. However, since we are now dealing with global-scale navigation, the base units will be different. We will be using a Local Navigation Frame, North-East-Down and so our base position units will be:

$$\mathbf{p} = \begin{bmatrix} \text{decimal degrees latitude} \\ \text{decimal degrees longitude} \\ \text{meters below mean sea level} \end{bmatrix} \tag{1}$$

$$\mathbf{p} = \begin{bmatrix} L \\ \lambda \\ h \end{bmatrix} \tag{2}$$

$$\mathbf{p} = \begin{bmatrix} \text{roll} \\ \text{pitch} \\ \text{yaw} \end{bmatrix} \tag{3}$$

will implement two architectures: a feed-forward and a feed-back INS. The feedback architecture is essentially the same as the previous nonlinear Kalman filter in Project 3. Therefore our state vector x for the **FeedBack Model** can be defined as:

$$x = \begin{bmatrix} L \\ \lambda \\ h \\ \phi \\ \theta \\ \psi \\ V_N \\ V_E \\ V_D \\ b_a x \\ b_a y \\ b_a z \\ b_g x \\ b_g y \\ b_g z \end{bmatrix}$$

The **Feed-Forward Model** will be:

$$x = \begin{bmatrix} L \\ \lambda \\ h \\ \phi \\ \theta \\ \psi \\ V_N \\ V_E \\ V_D \\ L-e \\ \lambda - e \\ h - e \end{bmatrix}$$

1.1 Propogation Model

The propagation (integrator) model for the IMU output is reiterated below. It must be done in three steps: attitude update, velocity update, and position update. It is necessary to maintain the prior (t-1) and the posterior state (t) belief throughout the calculation and return only the posterior.

1. **Attitude Update**
   Given the previous state x_{t-1} and the gyroscope measurements $\omega_i^b = [w_{\text{roll}} w_{\text{pitch}} w_{\text{yaw}}]^T$, if we use a feedback filter and modeling the IMU biases first subtract out the bias from the measured value ($\bar{\omega} = \omega_i^b - b_g$):

$$\omega_E = 7.2921157(10^{-5})\frac{\text{rad}}{\text{s}}(\text{Earth's rate of rotation}) \tag{4}$$

$$\Omega_i^e = \begin{bmatrix} 0 & -\omega_E & 0 \\ \omega_E & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{5}$$

$$\omega_e^n = \begin{bmatrix} \frac{v_E}{R_E(L)+h} \\ -\frac{v_N}{R_N(L)+h} \\ -\frac{v_e \tan L}{R_E(L)+h} \end{bmatrix} \tag{6}$$

$$\Omega_e^n = [\omega_e^n \Lambda] \tag{7}$$

$$\Omega_i^b = [\omega_i^b \Lambda] \tag{8}$$

$$R_{b,t}^n \approx R_{b,t-1}^n (I_3 + \Omega_i^b \delta t) - (\Omega_{e,t-1}^n + 2\Omega_{i,t-1}^e) R_{b,t}^n \delta t \tag{9}$$

2. **Velocity Update**

$$f_n^t \approx \frac{1}{2}(R_{b,t-1}^n + R_{b,t}^n) f_{b,t} \tag{10}$$

$$v_{n,t} = v_{n,t-1} + \delta t(f_{n,t} + g(L_{t-1}, h_{t-1}) - \Omega_{e,t-1}^n + 2\Omega_{i,t-1}^e) v_{n,t-1} \tag{11}$$

   where $g(L_{t-1}, h_{t-1})$ is the Somigliana gravity model.

3. **Position Update**

$$h_t = h_{t-1} - \frac{\delta t}{2}(v_{D,t-1} + v_{D,t}) \tag{12}$$

$$L_t = L_{t-1} + \frac{\delta t}{2}\left(\frac{v_{N,t-1}}{R_N(L_{t-1}) + h_{t-1}}\right) + \frac{\delta t}{2}\left(\frac{v_{N,t}}{R_N(L_{t-1}) + h_t}\right) \tag{13}$$

$$\lambda_t = \lambda_{t-1} + \frac{\delta t}{2}\left(\frac{v_{E,t-1}}{R_E(L_{t-1}) + h_{t-1}\cos L_{t-1}}\right) + \frac{\delta t}{2}\left(\frac{v_{E,t}}{R_E(L_{t-1}) + h_t \cos L_{t-1}}\right) \tag{14}$$

2 Project Implementation

2.1 TASK I: Observation Model

We will have two observation models defined for "FeedForward Model" and "FeedBack Model" individually. Given $\mathbf{x}$ as the current state vector, the observation matrix $\mathbf{H}$ is calculated as follows:

1. For the Feed-Forward method:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}$$

Where:

$\mathbf{I}_{3\times3}$ represents the identity matrix of size 3×3.
$\mathbf{0}_{3\times3}$ represents the zero matrix of size 3×3.

Finally, the observation model can be defined as:

$$\mathbf{z} = \mathbf{H}\mathbf{x} + \eta(\mathbf{0}, \mathbf{R})$$

$$z = \begin{bmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} L \\ \lambda \\ h \\ \phi \\ \theta \\ \psi \\ V_N \\ V_E \\ V_D \\ L-e \\ \lambda-e \\ h-e \end{bmatrix} + \eta(0, R)$$

2. For the Feed-Back method:

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} \end{bmatrix}$$

Where:

$\mathbf{I}_{3\times3}$ represents the identity matrix of size 3×3.
$\mathbf{0}_{3\times3}$ represents the zero matrix of size 3×3.

the observation model can be defined as:

$$\mathbf{z} = \mathbf{H}\mathbf{x} + \eta(\mathbf{0}, \mathbf{R})$$

$$z = \begin{bmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} L \\ \lambda \\ h \\ \phi \\ \theta \\ \psi \\ V_N \\ V_E \\ V_D \\ b_a x \\ b_a y \\ b_a z \\ b_g x \\ b_g y \\ b_g z \end{bmatrix} + \eta(0, R)$$

2.2 Non-Linear Kalman Filter

We will implement Unscented Kalman FIlter to complete Task II and Task III. The steps are similar to EKF, including defining the Predict Step, Update Step, and Observation Model. Following are few of the essential steps that should be included in Predict and Update step.

1. **Compute Sigma Points**
   Given the mean state vector $\mathbf{x}$ and covariance matrix P, generate sigma points $\mathcal{X}_i$ as follows:

$$\mathcal{X}_i = \begin{cases} \mathbf{x}, & \text{if } i = 0 \\ \mathbf{x} + \sqrt{(n+\kappa)P}, & \text{if } i = 1, ..., n \\ \mathbf{x} - \sqrt{(n+\kappa)P}, & \text{if } i = n+1, ..., 2n \end{cases}$$

2. **Compute Weights**

$$l = \alpha^2 \cdot (n + \kappa) - n$$
$$\text{Mean Weights}[i] = \frac{1}{2(n+l)}, \quad i = 0, 1, ..., 2n$$
$$\text{CovarianceWeights}[i] = \frac{1}{2(n+l)}, \quad i = 0, 1, ..., 2n$$
$$\text{MeanWeights}[0] = \frac{l}{n+l}$$
$$\text{CovarianceWeights}[0] = \frac{l}{n+l} + (1 - \alpha^2 + \beta)$$

3. **Correct Covarince**
   To correct covariance, we will define a correctCovar() function to ensure that a covariance matrix remains symmetric and positive definite. If the input covariance matrix is not symmetric or not positive definite, it symmetrizes the matrix, adjusts negative eigenvalues by adding a small constant j, and recursively calls itself with an increased j value until the resulting matrix is both symmetric and positive definite.

UKF is implemented for both Feed-Forward as well as Feed-Back Model. The Feed-Forward model returns a 12-state vector while the Feed-Back Model returns a 15-state vector.

2.2.1 TASK II: Non-Linear Error State Implementation

For the error state, we will initialize our UKF model with the dimensions of the state vector as 12. This is also known as the Feed-Forward Model.

Refer to the following plots for visualization. We will define the following parameters:
State Covariance Matrix, P : $I_{12\times12} * 1$
State Covariance Matrix, Q : $I_{12\times12} * 0.001$
State Covariance Matrix, R : $I_{12\times12} * 0.1$
kappa : 1.0
alpha : 1.0
beta : 0.4
Gyroscope Bias : $[1, 1, 1]$
Accelerometer Bias : $[1, 1, 1]$

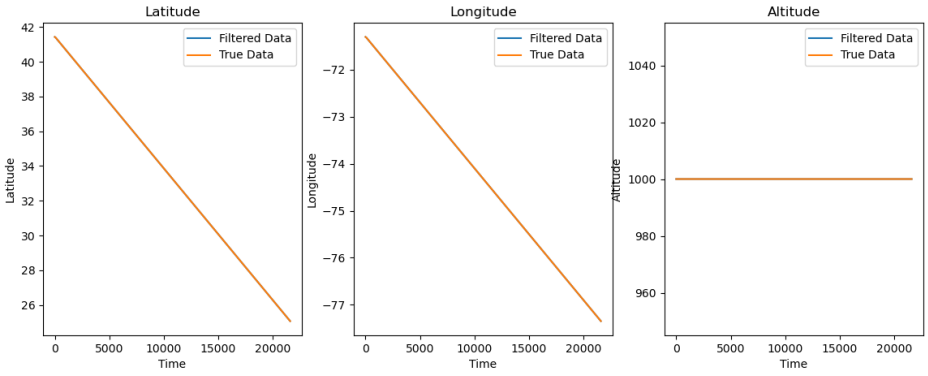

Figure 1: Position vs Time

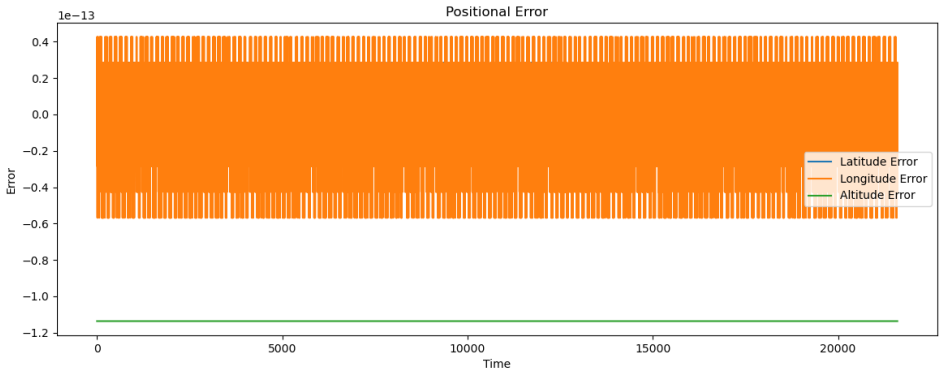

Figure 2: Position Error

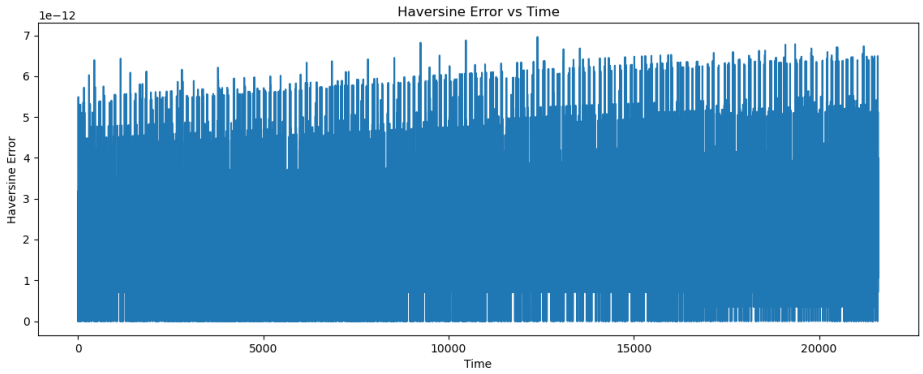

Figure 3: Haversine Error

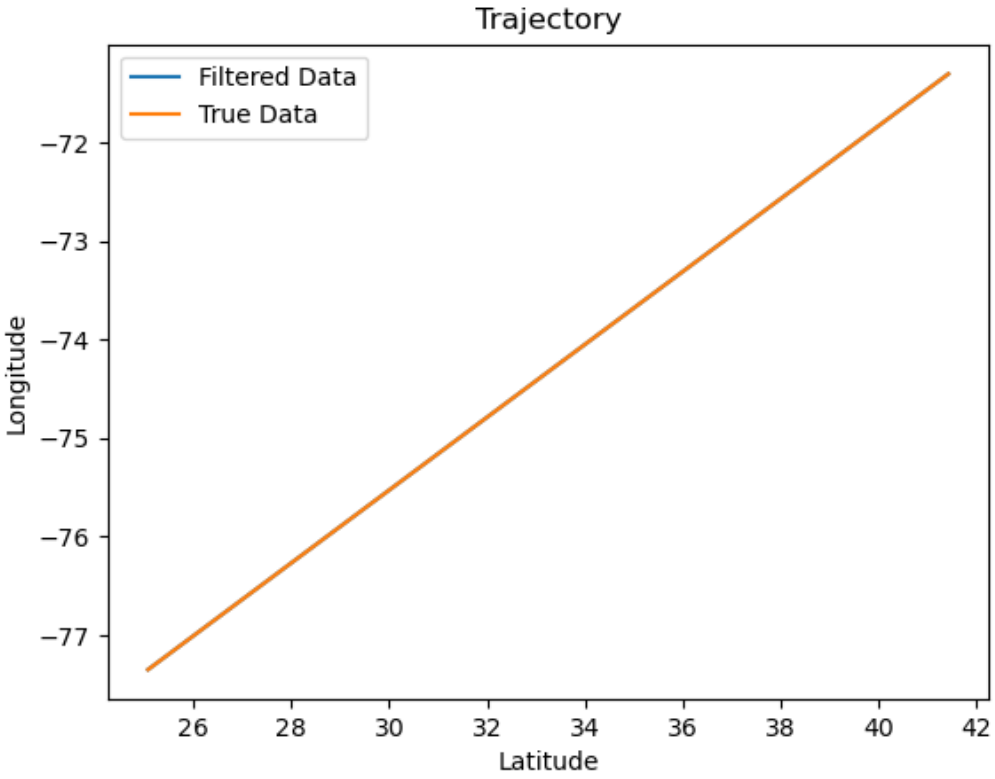

Figure 4: 2D Latitude vs Longitude

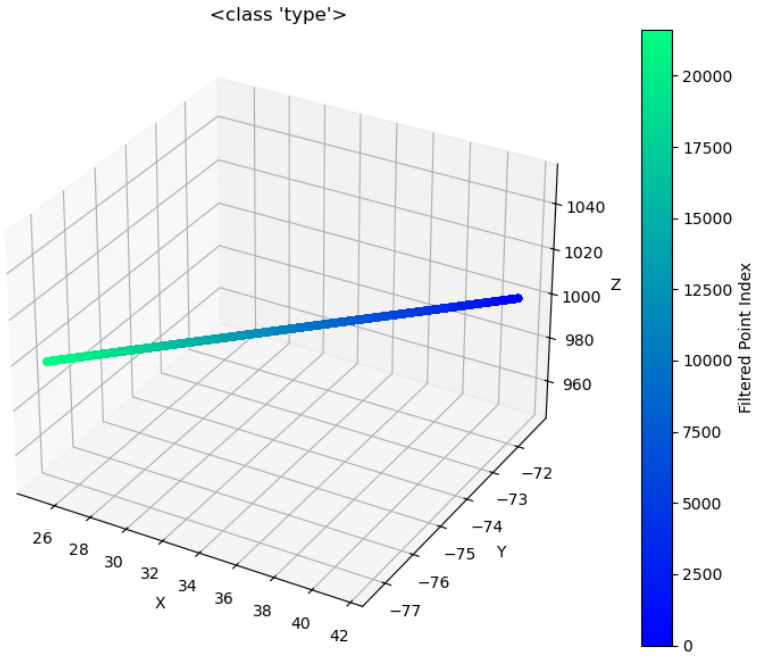

Figure 5: 3D Trajectory

2.2.2 Task III: Non-Linear Full State Implementation

For the error state, we will initialize our UKF model with the dimensions of the state vector as 15. This is also known as the Feed-Back Model.

Refer to the following plots for visualization. We will define the following parameters:
State Covariance Matrix, P : $I_{12\times12} * 5$
State Covariance Matrix, Q : $I_{12\times12} * 0.01$
State Covariance Matrix, R : $I_{12\times12} * 0.01$
kappa : 1.0
alpha : 1.0
beta : 0.4
Gyroscope Bias : $[1, 1, 1]$
Accelerometer Bias : $[1, 1, 1]$

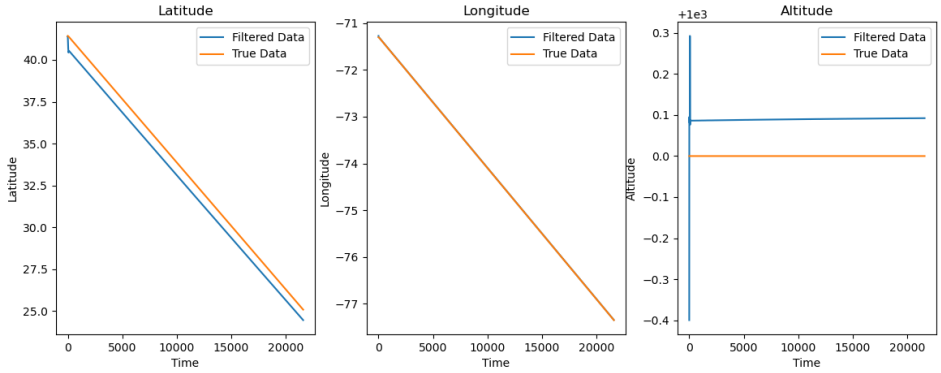

Figure 6: Position vs Time

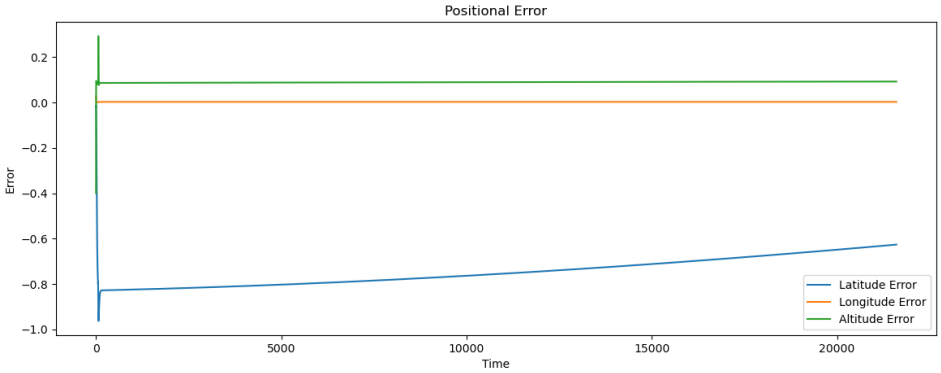

Figure 7: Position Error

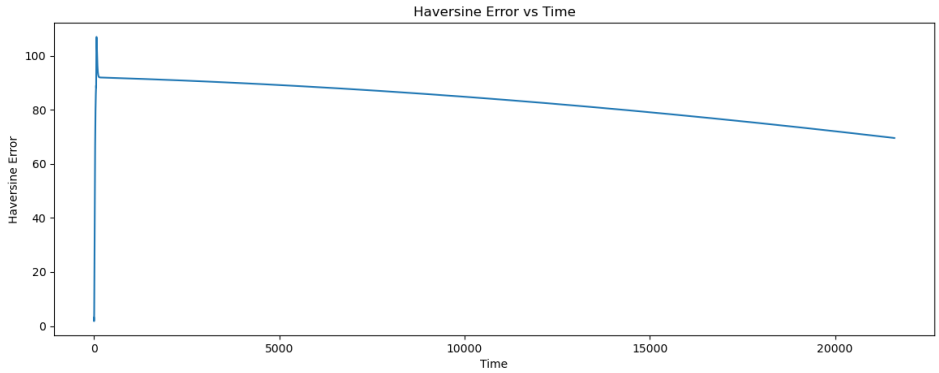

Figure 8: Haversine Error

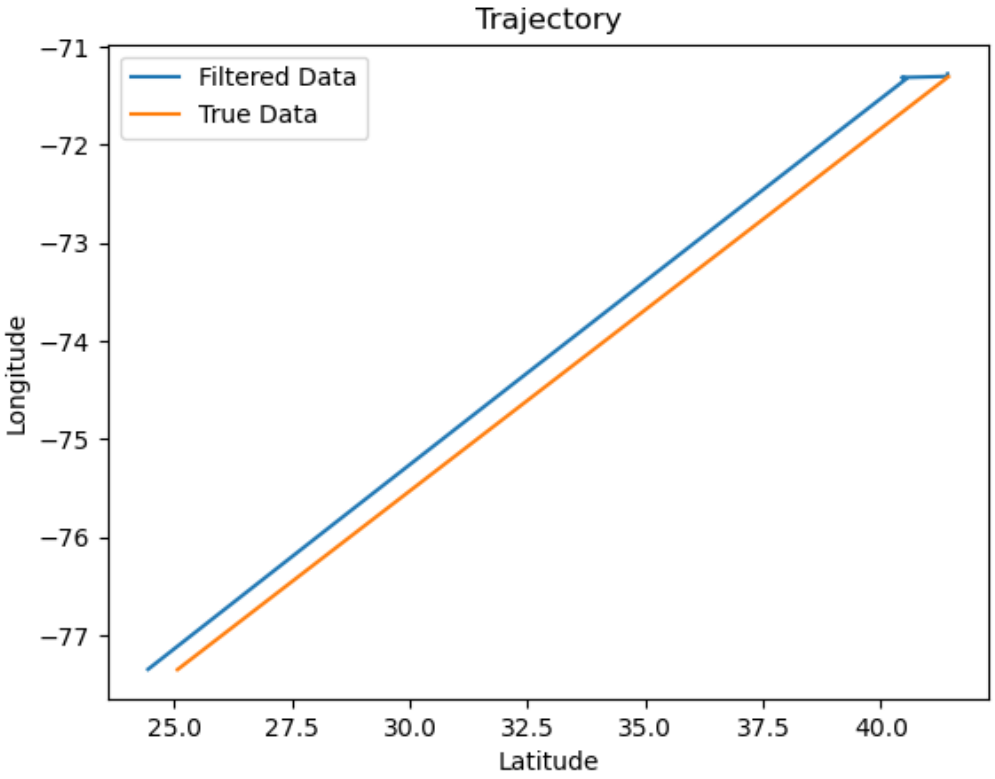

Figure 9: 2D Latitude vs Longitude

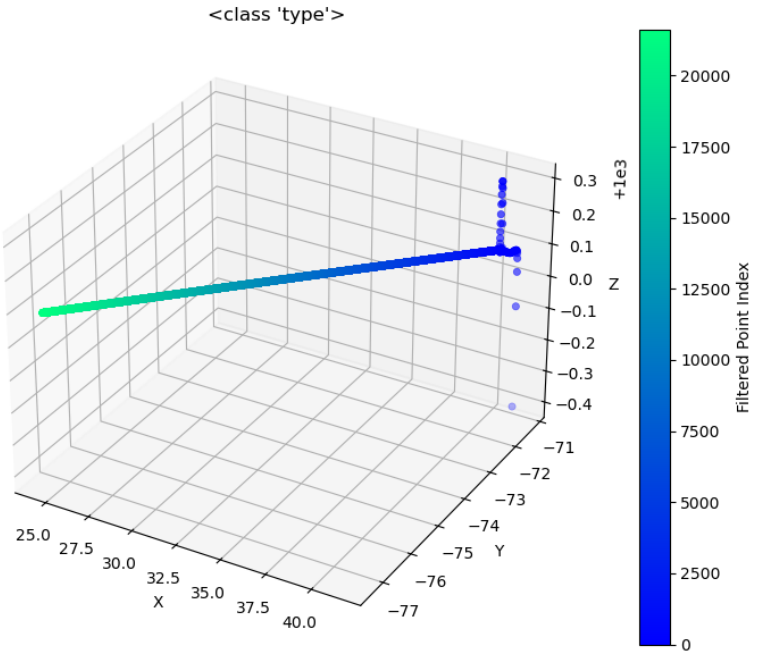

Figure 10: 3D Trajectory

2.3 TASK IV: Discussion and Performance Analysis

2.3.1 Error Analysis

We will analyze two different types of errors. **Generic Error** which typically refers to the difference between an observed or calculated value and the true or expected value. **Haversine error**, on the other hand, specifically relates to the inaccuracies that arise when using the Haversine formula to calculate distances between two points on a sphere, such as the Earth. The Haversine formula is commonly used in navigation and geographic information systems (GIS) to compute the great-circle distance between two points given their latitude and longitude coordinates. However, due to the Earth's slightly oblate shape and other factors, the Haversine formula introduces some error in distance calculations, especially over long distances. This error is referred to as Haversine error.

In the implementation, we found out that the Feed Back model performed better which can be observed from the Postional Error and Hvaersine Error plots in **Fig. 7** and **Fig. 8**. The plots appear to be stable and converging to 0 over Time. This is not the case so for Feed Forward model, as the Postional Error and Hvaersine Error plots in **Fig. 2** and **Fig. 3** are extremely noisy and do not converge to zero over time.

2.3.2 Was UKF a clever choice?

In assignment 3, we were dealing with a much smaller observation model prone to fewer non-linearities. Therefore, EKF was a better choice. However, in this assignment, we were dealing with a larger observation model with much more anticipated non-linearities. Implementing UKF, therefore, feels like a better choice. Pros and Cons that were observed in project implementation.

1. **Pros**:
   (a) UKF is particularly effective for estimating the state of nonlinear dynamic systems, where the traditional Extended Kalman Filter (EKF) may struggle due to linearization errors.
   (b) Unlike the EKF, which linearizes the nonlinear system model at each time step, the UKF approximates the distribution of the state directly using a set of carefully chosen sigma points. This avoids the need for complex and computationally expensive Jacobian matrix calculations.
2. **Cons**:

(a) The computation of sigma points, their propagation through the system model, and the subsequent calculation of mean and covariance estimates is computationally expensive.

(b) The performance of the UKF can be sensitive to the selection of parameters such as the scaling factor κ, the tuning parameters ($\alpha and \beta$) and the regularization constant j used in covariance correction.