ODTÜ METU

Aerospace Engineering Department

AE 484

Inertial Navigation Systems

Term Project

Comparison of Integrated Sensor Fusion Architectures for Autonomus Racing Drones

Instructor: Prof. Dr. Halil Ersin SÖKEN

Date: June, 2024

By: Team 3

Abdullah Hububati – 2491884

Arif Metrcan Koca – 2701696

Syed Yasir Rehan Ali – 2595676

Abstract

This paper demonstrates how to implement various Kalman filter architectures, namely a tightly-coupled extended Kalman filter, a loosely-coupled linear Kalman filter, and a tightly-coupled unscented Kalman filter, for estimating the position, velocity, and attitude of an autonomous racing drone. Inertial measurement unit measurements for specific force were used in conjunction with ultra-wide band pseudo-range and pseudo-range rates to predict and update the filters, respectively, for position and velocity estimation. Hence, this is an integrated navigation problem that uses inertial navigation system and aiding sensor measurements for drone navigation purposes. The UWB measurements were simulated using the ground truth translational position. Body angular rates from the IMU and quaternion measurements from a visible-light camera, equipped with a high-fidelity video processing system, were used to predict and update the filters, respectively, for attitude estimation. The high-fidelity video processing system provided the ground truth quaternion vector as well. The dataset pertains to the UZH-FPV's drone racing, visual-inertial odometry database.

Keywords: Tightly-Coupled, Extended Kalman Filter (EKF), Loosely-Coupled, Linear Kalman Filter, Unscented Kalman Filter (UKF), Inertial Measurement Unit (IMU), Ultra-Wideband (UWB), Integrated Navigation, Inertial Navigation System (INS), Drone Navigation, UZH-FPV

Table of Contents

I. Abbreviations 1

II. Notations 2

III. Introduction 3

III.I. Literature Survey 3

III.II. Methodology 4

IV. Dataset 5

V. Filters 10

V.I. OL TC EKF 11

V.II. OL LC LKF 15

V.III. OL TC UKF 18

VI. Conclusion 23

VII. References 24

I. Abbreviations

AHRS	Attitude Heading and Reference System
CL:	Closed-Loop
DCM:	Direction Cosine Matrix
EKF:	Extended Kalman Filter
GDOP:	Geometric Dilution of Precision
GT:	Ground Truth
HFVPS:	High-Fidelity Video Processing System
IMU:	Inertial Measurement Unit
INS:	Inertial Navigation System
KF:	Kalman Filter
LC:	Loosely-Coupled
LKF:	Linear Kalman Filter
LLNF:	Local-Level Navigation Frame
NED:	North, East, Down
NLS:	Nonlinear Least Square
OL:	Open-Loop
TC:	Tightly-Coupled
TSE:	Taylor Series Expansion
UKF:	Unscented Kalman Filter
UWB:	Ultra-Wideband
UZH-FPV:	University of Zurich-Flight Performance Visualizer

II. Notations

- **Scalar Variables**: Lower- or upper-case letter or symbol (e.g., κ).

- **Vectorial Variables:** Lower- or upper-case letter with arrow accent:
  - **Position Vectors:** Lower-case 'r' (i.e., $\vec{r}$).
  - **Velocity Vectors:** Upper-case 'v' (i.e., $\vec{V}$).
  - **Quaternion Vectors:** Lower-case 'q' (i.e., $\vec{q}$),
    - Scalar followed by vectorial quaternions (i.e., $\begin{bmatrix} q_w & \vec{Q}_{xyz}{}^T \end{bmatrix}^T$).
  - **State Vectors:** Lower-Case 'x' (i.e., $\vec{x}$).
  - **Input Vectors:** Lower-Case 'u' (i.e., $\vec{u}$).
  - **Measurement Vectors:** Lower-case 'y' (i.e., $\vec{y}$).

- **Measured Variables:** Variables with Tilda accent (e.g., $\widetilde{\vec{\omega}}$).

- **Matrices:** Upper-Case letter in bold with square brackets:
  - **Identity Matrix:** Upper-Case 'i' with right subscript number (e.g., $[\boldsymbol{I_3}]$ is a 3x3 identity).
  - **Attitude Matrix:** Upper-Case 'c' with right subscript letters (e.g., $[\boldsymbol{C_{nb}}]$ is body to LLNF).
  - **State Propagation Matrix:** Upper-Case 'a' (i.e., $[\boldsymbol{A}]$).
  - **State Propagation Matrix:** Upper-Case 'b' (i.e., $[\boldsymbol{B}]$).
  - **State Covariance Matrix:** Upper-Case 'p' (i.e., $[\boldsymbol{P}]$).
  - **Process Noise Covariance Matrix:** Upper-Case 'q' (i.e., $[\boldsymbol{Q}]$).
  - **Measurement Noise Covariance Matrix:** Upper-Case 'r' (i.e., $[\boldsymbol{R}]$).
  - **Measurement Weight Matrix:** Upper-Case 'w' (i.e., $[\boldsymbol{W}]$).
  - **Kalman Gain Matrix:** Upper-Case 'k' (i.e., $[\boldsymbol{K}]$).
  - **(Jacobian of) State-to-Measurement Matrix:** Upper case 'h' in bold (i.e., $[\boldsymbol{H}]$),
    - Jacobian numerically determined at a state (e.g., $[\boldsymbol{H}|_{\vec{x}_k^-}]$).
  - **Jacobian Nonlinear Least Square Matrix:** Upper case 'g' in bold (i.e., $[\boldsymbol{G}]$).

- **Reference Frames:** Right subscript or superscript letter(s):
  - **Inertial Frame:** Lower-Case 'i'.
  - **Body Frame:** Lower-Case 'b'.
  - **LLNF:** Lower-Case 'n'.
  - **Inertial LLNF:** Upper-Case 'n' (i.e., N).
  - **Relative to/W.R.T.:** Lower-Case right subscript letters (e.g., $\vec{\omega}_{ib}$ is in body w.r.t. inertial).
  - **Resolved in:** Lower-case right superscript letter (e.g., $\vec{\omega}^b$ is resolved in body).

- **Predicted Vector/Matrix:** right '-' superscript (e.g., $\vec{x}_k^-$).

- **Updated Vector/Matrix:** right '+' superscript (e.g., $\vec{x}_k^+$).

III. Introduction

This paper provides a comprehensive comparison of three different OL KFs for estimating the navigational solution (i.e., position, velocity, and attitude) of autonomous drones. The objective of this comparison is to determine the optimal sensor fusion architecture for low-grade IMUs that provides an appropriate tradeoff between computational load and complexity and solution accuracy and reliability. Such a determination would benefit the field of autonomous racing drones by allowing platforms to effectively navigate around racing courses in the required manner with minimal intervention. Accordingly, we conduct a literature survey to establish some expectations for our results and formulate a methodology.

III.I. Literature Survey

A recent thesis[1] dealt with implementing an LKF and an EKF to estimate the full state (i.e., triaxial position, velocity, and attitude) of an UAV. The thesis discussed that while the LKF is an optimal estimator with virtually no drawbacks to its application, it cannot process a nonlinear mathematical model for the prediction or a nonlinear measurement model for the update. Conversely, an EKF sacrifices the filter's optimality to be able to process such nonlinear modelling scenarios. The EKF realizes this by using a 1st-order TSE linearization on the nonlinear models to reformulate the navigation problem into a linear, albeit low-order, approximate counterpart before proceeding to apply the LKF process. This results in the filter ignoring the errors due to the TSE approximations during both of its overarching steps, which introduces suboptimality.

In fact, it was discussed in the same thesis[1] that a 1st-order TSE approximation is only accurate in the event that the higher-order terms are negligibly small in comparison. However, for a high-speed, 6 degree of freedom UAV application, this is not the case, which introduces significant biases into the filter estimates.

On the other hand, a different thesis[2] discussed that a UKF can bypass the introduction of approximation errors into the filter by approximating the Gaussian probability distribution of the states themselves without attempting to linearize them. This approximation enables the UKF to utilize the original, nonlinear models to propagate and update the state, resulting in typically much more accurate filter estimates compared to a conventional EKF since no explicit errors are introduced into the system without being accounted for by the covariance.

The same thesis[2] outlined the process of computing so called "Sigma Points", exclaiming that since the EKF processes the Gaussian distribution nonlinearly, it robs it of its inherent properties that constitute it into a Gaussian distribution. On the other hand, the computation of sigma points allows the distribution to retain its Gaussian nature to better capture the covariance of highly nonlinear motion profiles. This technique is known as the "Unscented Transformation", and it computes points in the neighborhood of Gaussian distribution to facilitate an improvement in the filter estimation's accuracy and robustness.

III.II. Methodology

First, a dataset relevant to drone racing was procured from the UZH-FPV database for visual-inertial odometry. The dataset included IMU and GT measurements. Next, aiding sensor data, namely UWB pseudo-range and pseudo-range rate measurements, were simulated using GT data and a local level navigational problem was formulated using appropriate assumptions. Then, in-line with the literature survey, an OL TC EKF sensor fusion architecture was implemented to integrate the IMU and simulated aiding sensor measurement, which was the most widely used architecture for similar purposes. The EKF was tuned to ensure equivalent levels of covariance for both used measurements and an appropriate matching between the filter estimate and the ground truth measurements. Finally, two alternative OL sensor fusion architectures, namely a LC KF and a TC UKF, were tested against the dataset using the same covariance values.

The two primary challenges faced throughout the implementation of the methodology pertained to the lengthy EKF tuning process and rectification of the GDOP impeding on the position estimates in the z-axes, particularly for the UKF. Regarding the former, the particularly low-grade IMU used by the platform made it so that the prediction step for all KFs resulted in highly inaccurate state prediction due to biases above all else, such that an extensive and rigorous tuning procedure was necessary to establish an appropriate balance between the process and measurement noise covariance matrices. Else, the EKF, and hence the two alternative filters, estimated the state vector with an extensive drift that would require a CL system to otherwise rectify and remedy.

On the other hand, the effect of the GDOP phenomenon in an indoor environment meant that the placement of the UWB transmitters would have to take into account that excessive proximity between the transmitters with respect to current drone position could potentially result in the deletion of one, or more, pseudo UWB measurements. In fact, the effect of the GDOP on the UKF was so prominent that despite placing four different UWB transmitters as far apart as reasonably possible in a horizontal (xy) plane containing the origin, a fifth UWB transmitter with a different z-axis position had to be included to allow the UKF to even begin to capture the intricacies of the quick, highly nonlinear drone motion in that axis.

IV. Dataset

The dataset used within the context of this paper belongs to the UZH-FPV's indoor, forward-facing race track #10[4]. This was an indoor race track and the measurements were collected using the Snapdragon experimental setup, which enabled the collection of both IMU and GT measurements.

The IMU data were composed of 38,137 measurements for two three-dimensional vectors, six quantities overall, sampled at 500 [Hz]. These vectors were the specific force in body frame '$\vec{f}_b$' and the body angular rates with respect to inertial frame, resolved in body frame, '$\vec{\omega}_{ib}^{b}$'. The IMU used was InvenSense's MPU-9250, which includes a magnetometer, making it an AHRS, though magnetometer measurements were not sampled.

On the other hand, the GT data were composed of 14,950 measurements for two vectorial quantities, namely the translational position with respect to an LLNF '$\vec{r}_n$' and the attitude's quaternion with respect to the same LLNF '$\vec{q}_n$'. Hence, overall, seven quantities were measured, again at a sampling frequency of 500 [Hz].

Due to the matching of the sampling frequency between the IMU and GT measurements, they were time synchronized measurements, though the IMU began sampling measurements before, and continued sampling measurements after the GT sensors. Thus, only the portion of the IMU measurements with corresponding GT measurements were considered for our analyses.

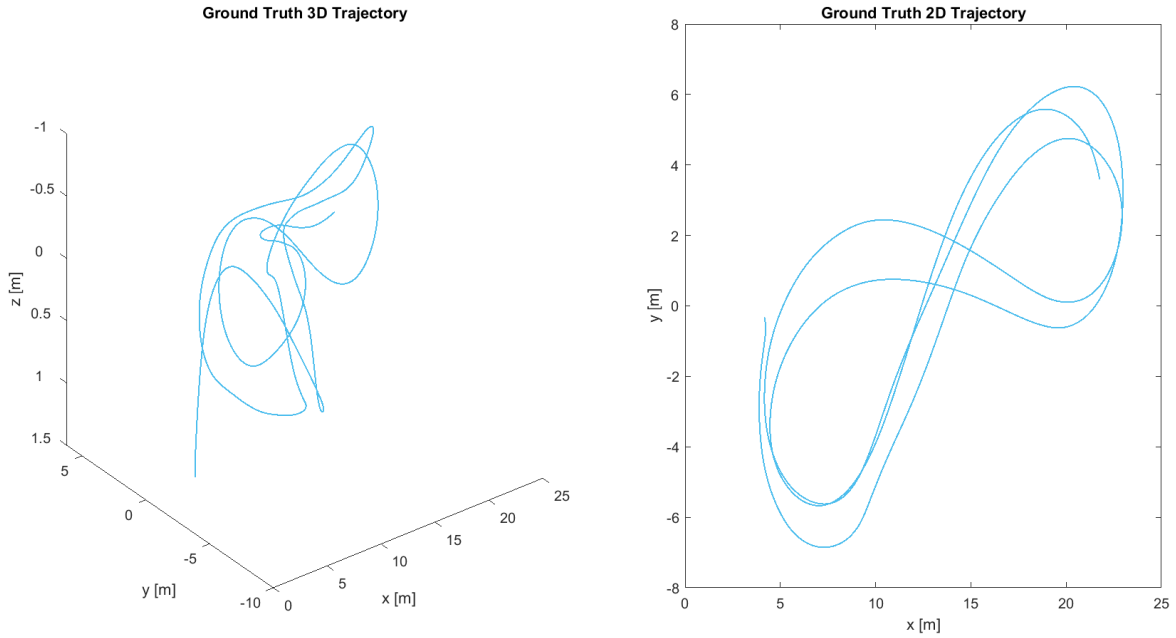
The GT measurements were used to simulate the UWB pseudo-range and pseudo-range rate measurements for the aiding sensor measurements necessary for a position and velocity state estimate. The UWB measurements were simulated using Eqns. 4.1 & 4.2 below, which require knowledge of the GT drone position and velocity and the kth UWB transmitters' position and velocity.

$$\rho_k = \sqrt{(x - x_k)^2 + (y - y_k)^2 + (z - z_k)^2} \tag{4.1}$$

$$\dot{\rho}_k = \frac{(x - x_k) * (V_x - V_{x,k}) + (y - y_k) * (V_y - V_{y,k}) + (z - z_k) * (V_z - V_{z,k})}{\rho_k} \tag{4.2}$$

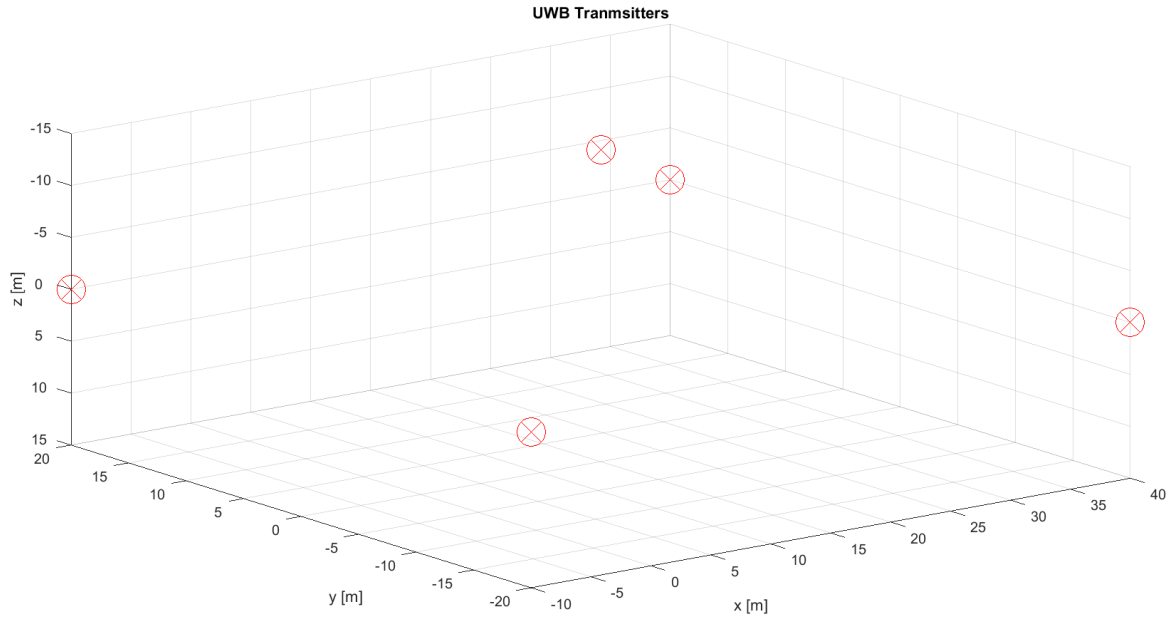
Most UWB transmitters are fixed, such that $\vec{V}_k = \vec{0}$, which simplifies Eq. 4.2. Additionally, trilateration requires having at least three different pseudo-range measurements at any moment to obtain a position navigational solution. Thus, we instead use five transmitters, which allows the filters the chance to estimate, and hence eliminate, some of the measurement-associated errors. These UWB transmitters were placed as shown in Fig. 4.2, according to the perceived parameters of the closed environment based on the GT trajectory shown in Fig. 4.1 below. Moreover, UWB measurements will be sampled 100 times slower compared to GT and IMU data.

Fig. 4.1: GT Drone Trajectory.



From Fig. 4.1, we deduce from the inversion of the z-axis of the drone's position trajectory that the LLNF used by GT measurements is a NED frame.

Fig. 4.2: UWB Transmitters' Placement.



In the NED frame whose origin is at the geometric center of the left wall of the testing environment, the UWB transmitters were placed along the walls, such that four were placed at the corners of a horizontal (xy) plane at zero elevation, while the last was placed dead-center along the ceiling.

Having found the correct definition of the LLNF which the GT measurements are given with respect to, we make use of that frame for the mechanization of the IMU measurements by assuming that this NED LLNF is inertial (i.e., star-fixed). Thus, we employ the mechanization equations in NED frame under the following assumptions, which yields the three mechanization equations Eqns. 4.3, 4.4, & 4.5:

- GT data is in NED LLNF, so we assume that frame to be inertial (i.e., $[\boldsymbol{D}]^{-1} = [\boldsymbol{I_3}]$ & $i = n = N$).
- Total duration is under 30 [s], so Earth's rotation is negligible (i.e., $\vec{\omega}_{ie}^{n} = \vec{0}$).
- Maximum displacement is under 30 [m], so Earth's curvature is negligible (i.e., $\vec{\omega}_{en}^{n} = \vec{0}$).

$$[\dot{\boldsymbol{C}_{\boldsymbol{Nb}}}] = [\vec{\omega}_{Nb}^{b} \times][\boldsymbol{C_{Nb}}] \tag{4.3}$$

$$\dot{\vec{V}}_N = [\boldsymbol{C_{Nb}}]\vec{f}_N + \vec{g}^n \tag{4.4}$$

$$\dot{\vec{r}}_N = V_N \tag{4.5}$$

The data processing algorithm is outlined in Alg. 4.1 below.

Alg. 4.1: Data Processing Algorithm.

1. *Load Measurements from Dataset (IMU & GT)*:
   a. *Load GT Position and Quaternion Attitude w.r.t. LLNF.*
   b. *Load IMU Specific Force and Body Angular Rates w.r.t. Inertial Frame.*
2. *Simulate UWB Measurements (ρ & $\dot{\rho}$)*:
   a. *Simulate GT Velocity by Numerical Differentiation of GT Position.*
   b. *Simulate Pseudo Range Measurements with $\sigma_\rho = 5/3\ [cm]$* (Eq. 4.1).
   c. *Simulate Pseudo Range Rate Measurements with $\sigma_\rho = 5/3\ [cm/s]$* (Eq. 4.2).
3. *Mechanize IMU Measurements to Obtain INS State Measurements*:
   a. *Initialize State Rates using GT States.*
   b. *Propagate Attitude by Inertial LLNF DCM Kinematics* (Eq. 4.3).
   c. *Propagate Velocity by Inertial LLNF Mechanization* (Eq. 4.4).
   d. *Propagate Position by Inertial LLNF Mechanization* (Eq. 4.5)
4. *Simulate INS State Measurements using GT Data and IMU Errors.*
   a. *Simulate GT Euler Angular Rates by Numerical Differentiation of GT Attitude.*
   b. *Determine GT Body Angular Rates by Inverse Transformation.*
   c. *Simulate GT Acceleration by Numerical Differentiation of GT Velocity.*
   d. *Determine GT Specific Force by Subtracting Gravitational Acceleration.*
   e. *Set Accelerometer and Gyroscope Errors*[3].
   f. *Simulate IMU Measurements.*
   g. *Repeat 4.1.4 for Simulated IMU Measurements.*

In Alg. 4.1 above, 4.1.4 was a necessary step to judge whether the perceived drifts and errors in the mechanized INS states, with respect to the GT states, was within the expected boundaries imposed by the IMU sensor errors or not. The reason that necessitated this verification becomes apparent once Figs. 4.3, 4.4, & 4.5 below are examined.

Fig. 4.3: Drone Linear Position Time Variation.

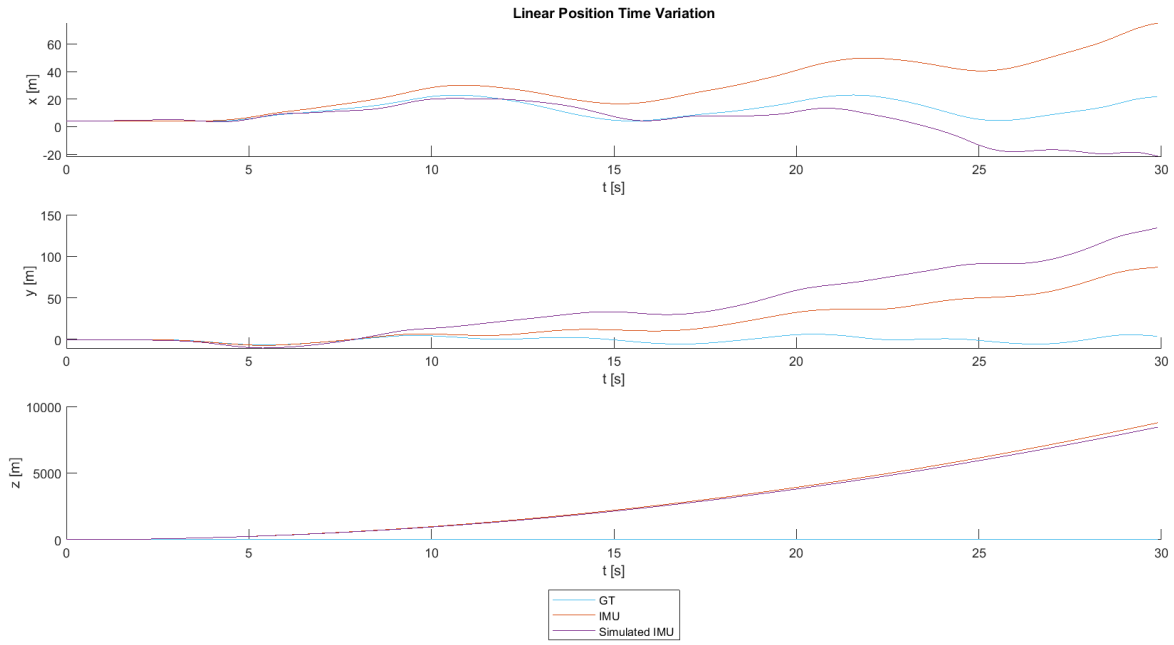


Fig. 4.4: Drone Linear Velocity Time Variation.

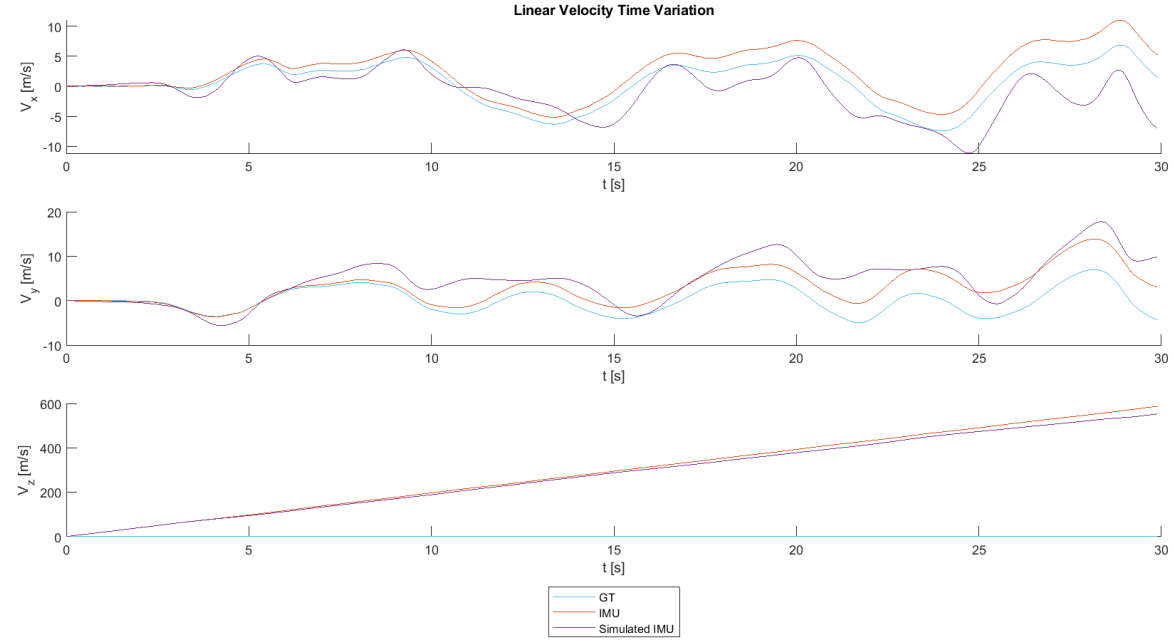
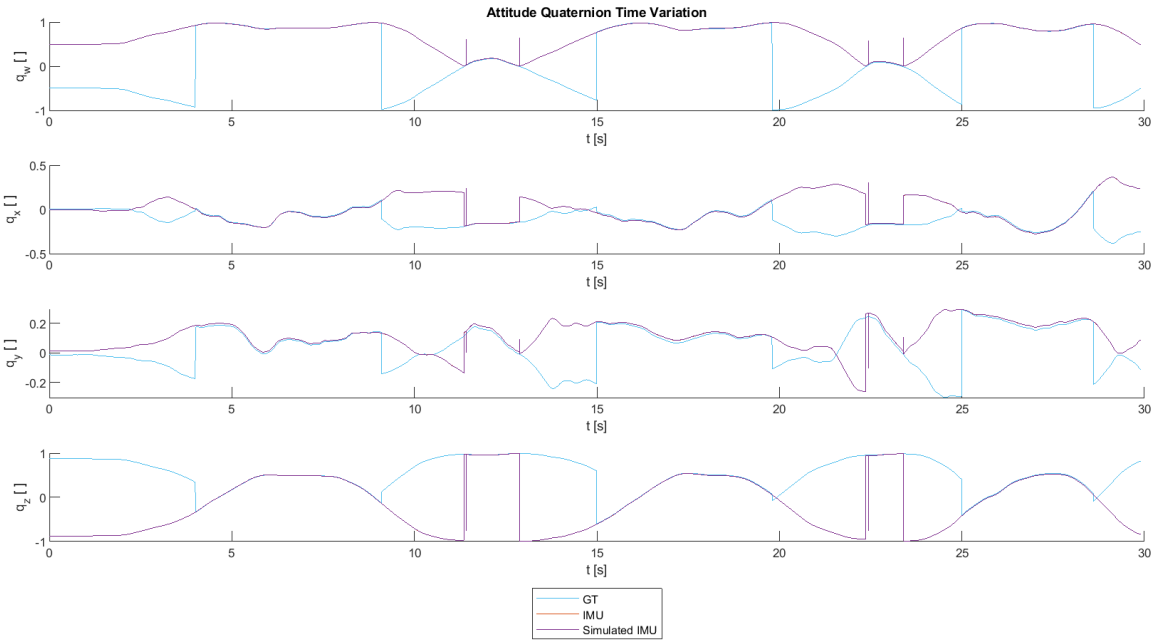


Fig. 4.5: Drone Attitude Quaternion Time Variation.



Obviously, from Fig 4.3, the mechanized, actual IMU measurements exhibit a substantial drift from the GT position measurements, particularly in the z-axis where the drift is in the order of 10 [km]. This drift obviously results from Eq. 4.5 and Fig. 4.4, such that the drift in z-axis velocity is much more prominent compared to the miniscule errors exhibited in the x- and y-axis velocities. This behavior cast a doubt on the correctness of the IMU measurements, particularly the specific force in z-axis '$\tilde{f}_{N,z}$'.

Additionally, examining Fig. 4.5 above, one can see that the quaternions derived from the actual IMU measurements are somehow, more-or-less, reflected images of the GT attitude about the zero-line, which further casts doubt regarding the body angular rates of the IMU measurements as a whole '$\widetilde{\bar{\omega}}_{Nb}^{b}$'.

Therefore, the most straightforward approach to confirm whether the doubtful IMU measurements are, in fact, correct interpretations of the GT measurements under the influence of sensor errors, we simulated the IMU measurements as in Alg. 4.1.4.

As can be seen from Figs. 4.3, 4.4, & 4.5, the simulated and actual IMU measurements resulted in very similar INS state estimates, meaning that the sensor biases were indeed the source of the perceived drifts.

V. Filters

Having properly formulated an indoor drone navigation problem, we may now move to designing the sensor fusion architecture that will combine INS state estimates, as a prediction for the KF state estimate, and aiding sensor measurements, as an update for the KF state estimate.

Overall, a comparison between an OL TC EKF, an OL LC LKF, and an OL TC UKF is in order. Hence, the block diagram flowcharts for those sensor fusion architectures, the details of their filter governing equations, and overall sensor fusion algorithm for all three filter architectures will be examined in detail. However, to be able to compare the performance of all three filters with one another, it is necessary to ensure the equivalence of the propagation and covariance matrices employed by all three filters. Thus, since the EKF is the most widely used architecture, the covariance matrices will be tuned with respect to the EKF results until an appreciable degree of accuracy is reached.

Fig. 5.1: Drone Trajectory Estimate Comparison.

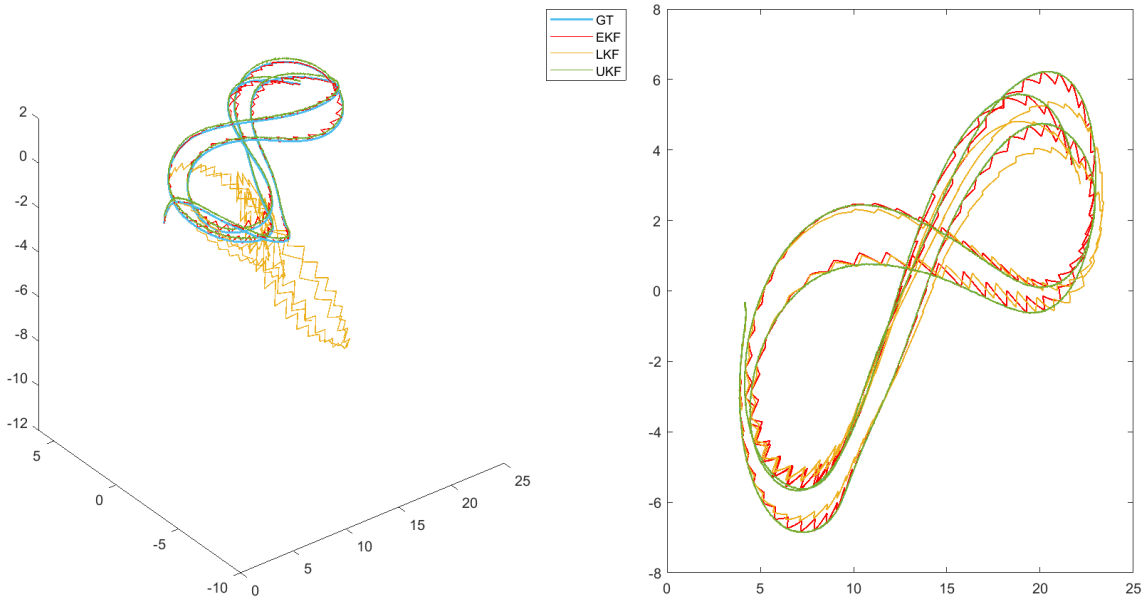


Fig. 5.1 above serves as a preliminary comparison for the performance of all three filters with respect to the GT trajectory. It clearly showcases that the UKF was superior in terms of estimation accuracy to the EKF, which was in turn superior to the LKF. These results are definitely consistent with our expectations.

The state vector to be estimated by all KFs is $\vec{x} = [\vec{P} \quad \vec{V} \quad \vec{V}]^T$ for position, velocity, and attitude. Also, the input vector used by all KFs is $\vec{u} = \begin{bmatrix} \tilde{\vec{a}}_{N,imu} & \tilde{\vec{q}}_{N,imu} \end{bmatrix}^T$, where $\tilde{\vec{a}}_{N,imu} = [\boldsymbol{C_{Nb}}]\tilde{\vec{f}}_{b,imu} - \vec{g}^n$. Hence, the state prediction employed in all KFs is $\vec{x}_k^- = [\boldsymbol{A}]\vec{x}_{k-1}^+ + [\boldsymbol{B}]\vec{u}_{k-1}$, where:

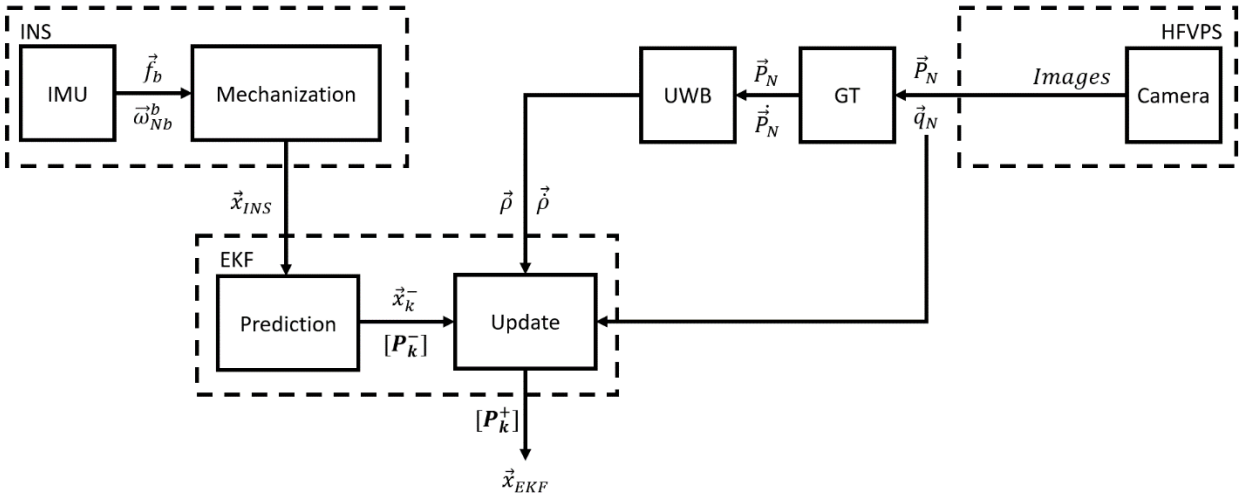
$$[\boldsymbol{A}] = \begin{bmatrix} [\boldsymbol{I_3}] & [\boldsymbol{I_3}](dt) \\ & [\boldsymbol{I_7}] \end{bmatrix}_{10\times10} \qquad [\boldsymbol{B}] = \begin{bmatrix} [\boldsymbol{I_3}](dt^2/2) & \\ & [\boldsymbol{I_7}](dt) \end{bmatrix}_{10\times7}$$

Finally, while all KFs work at a frequency of 500 [Hz], which is the IMU's sampling frequency, attitude update is only available every 10 steps, while UWB update is only available every 100 steps, such that the necessary adjustments were made to proceed without an update when unavailable.

V.I. OL TC EKF

First, the open-loop, tightly-coupled extended Kalman filter architecture will be examined. This architecture uses a nonlinear, and hence suboptimal, KF to estimate the state using mechanized IMU, or INS, state to predict the state and then using raw UWB measurements and camera quaternion measurements to update the state.

Fig. 5.2: OL TC EKF Architecture.



According to the above-shown sensor fusion architecture from Fig. 5.2, the governing equations for an OL TC EKF are Eqns. 5.1 & 5.2 below, where the measurement update is a 14x1 vector composed of five pseudo-range, five pseudo-range rate, and four quaternion measurements. Concurrently, by Eqns. 4.1 & 4.2, the relationship between the state vector '$\vec{x}$' and the measurement vector '$\vec{y}$' is nonlinear, spare for the attitude estimation part. Thus, it is necessary to use a nonlinear Kalman filter since the update block is nonlinear, as the calculation of the Kalman gain '$[\boldsymbol{K}]$' and state covariance '$[\boldsymbol{P}]$' matrices in that block require the Jacobian '$\left[\boldsymbol{H}|_{\vec{x}_k^-}\right]$'.

Prediction

$$\vec{x}_k^- = [\boldsymbol{A}]\vec{x}_{k-1}^+ + [\boldsymbol{B}]\vec{u}_{k-1}$$

$$[\boldsymbol{P}_k^-] = [\boldsymbol{A}][\boldsymbol{P}_{k-1}^+][\boldsymbol{A}]^{-1} + [\boldsymbol{Q}] \tag{5.1}$$

Update

$$[\boldsymbol{K}] = [\boldsymbol{P}_k^-]\left[\boldsymbol{H}|_{\vec{x}_k^-}\right]^T \left[\left[\boldsymbol{H}|_{\vec{x}_k^-}\right][\boldsymbol{P}_k^-]\left[\boldsymbol{H}|_{\vec{x}_k^-}\right]^T + [\boldsymbol{R}]\right]^{-1}$$

$$\vec{x}_k^+ = \vec{x}_k^- + [\boldsymbol{K}]\left(\vec{y}_k - h(\vec{x}_k^-)\right) \tag{5.2}$$

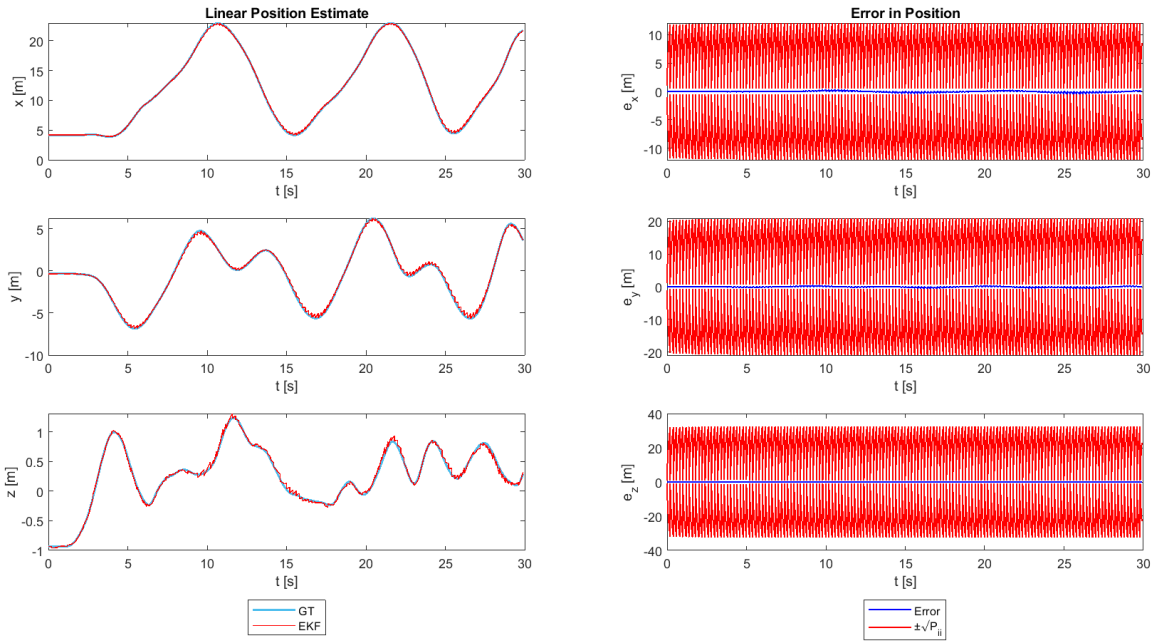
$$[\boldsymbol{P}_k^+] = \left[[\boldsymbol{I}_{10}] - [\boldsymbol{K}]\left[\boldsymbol{H}|_{\vec{x}_k^-}\right]\right][\boldsymbol{P}_k^-]$$

For this particular EKF, the Jacobian is as follows, where we note that the 4x4 identity matrix is used to map a linear relationship between the attitude quaternion prediction and update:

$$[\boldsymbol{H}] = \begin{bmatrix} \frac{x-x_1}{\rho_1} & \frac{y-y_1}{\rho_1} & \frac{z-z_1}{\rho_1} & & & \\ \vdots & \ddots & \vdots & & & \\ \frac{x-x_5}{\rho_5} & \frac{y-y_5}{\rho_5} & \frac{z-z_5}{\rho_5} & & & \\ \frac{V_x}{\rho_1} & \frac{V_y}{\rho_1} & \frac{V_z}{\rho_1} & \frac{x-x_1}{\rho_1} & \frac{y-y_1}{\rho_1} & \frac{z-z_1}{\rho_1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{V_x}{\rho_5} & \frac{V_y}{\rho_5} & \frac{V_z}{\rho_5} & \frac{x-x_5}{\rho_5} & \frac{y-y_5}{\rho_5} & \frac{z-z_5}{\rho_5} \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \\ \\ [\boldsymbol{I_4}]_{14\times 10} \end{matrix}$$

Application of this algorithm yields not only a state estimate and state covariance, but also an error with respect to the corresponding GT state. Hence, we compare the state estimate with the GT state and the error with the state covariance in Figs. 5.3, 5.4, & 5.5 below.

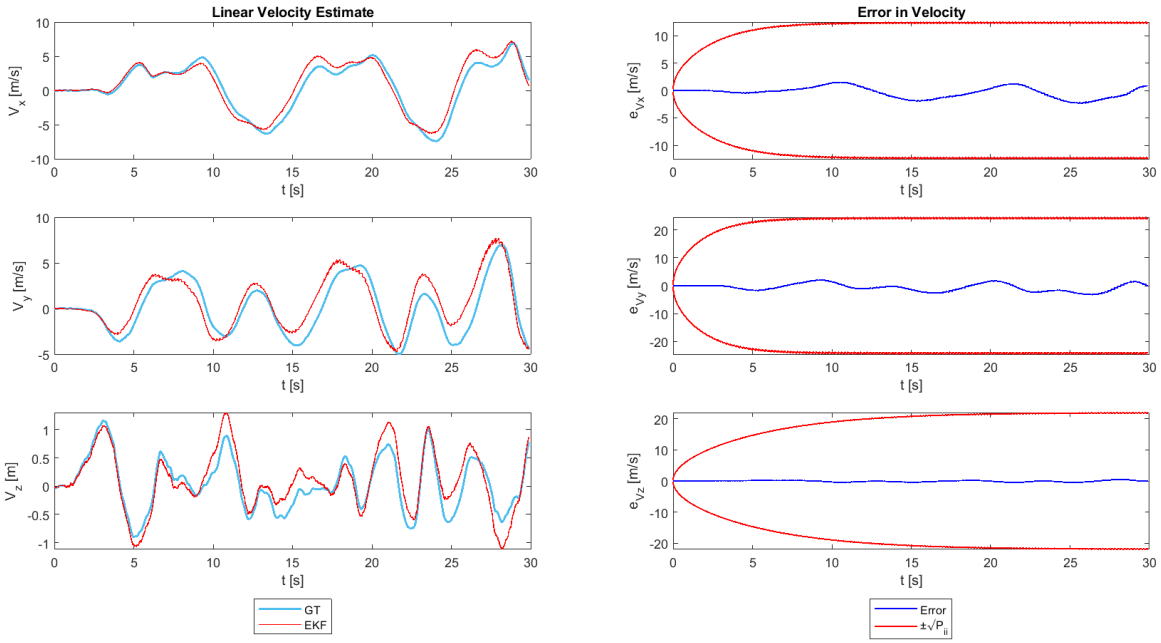
Fig. 5.3: OL TC EKF Position Estimate and Error.



The subplots on the left of Fig. 5.3 indicate that the OL TC EKF position estimate adequately captures the overall trajectory, though with the noise-like deviation from the actual value; a trademark characteristic of any KF estimation. Moreover, the subplots on the right show that the error is well-bounded within the respective covariance elements. This indicates that the EKF used is neither too optimistic nor too pessimistic, a sign of an effective tuning process. However, in regions of highly-nonlinear behavior, the EKF estimate seems to exhibit notable deviation from the GT state due to the Jacobian being but a 1st-order TSE approximation, especially in the z-axis.

Another point to note is that the position covariance exhibits many discontinuities, a behavior associated with a KF operating at a higher frequency compared to the measurement update sampling frequency. Under such circumstances, the covariance would linearly increase while no update is available, then sharply plummet once one becomes available. Thus, we may conclude that our manner of implementing the no-update adjustments was successful.

Fig. 5.4: OL TC EKF Velocity Estimate and Error.

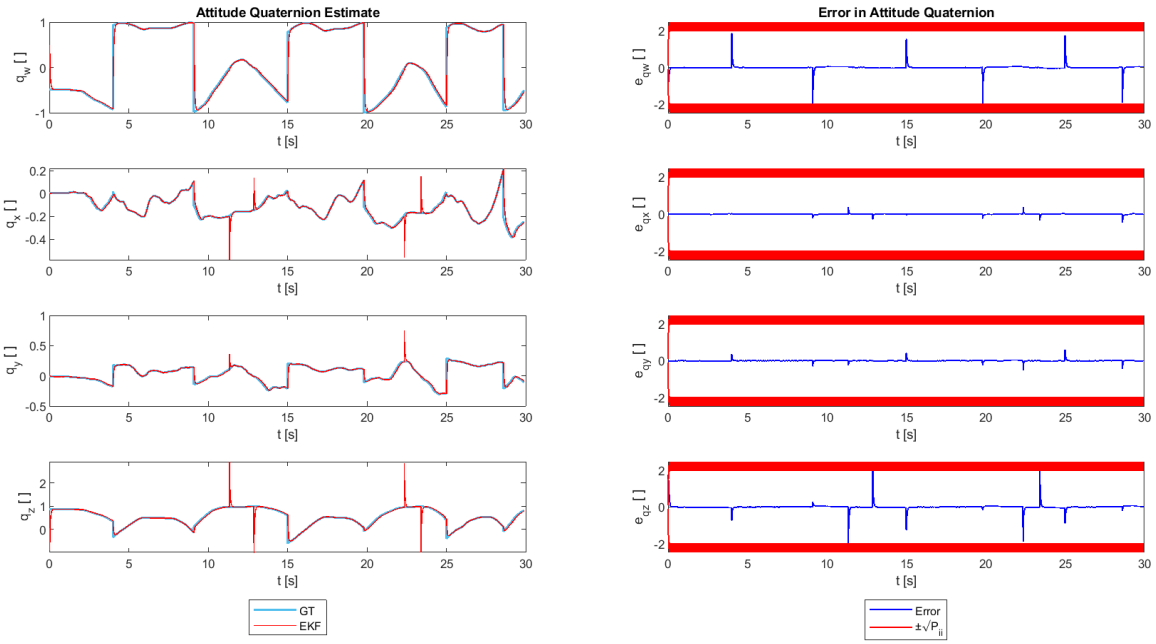


Contrary to the position estimate, the velocity estimate appears to be rather too pessimistic as there exists an appreciably large margin between the error and the covariance, meaning that the filter is relying more on the update and less on the prediction (i.e., Kalman gain is high). However, the magnitude of the error itself is quite low, just as it was for the position estimates. This is better showcased by the subplots on the left of Fig. 5.4 compared to those on the right, as the error is clearly on the order of magnitude of about 1 [m/s], give-or-take.

However, the true and estimated values for the velocity appear to be very distinct from one another, unlike the for position. Another point of contrast is that the high-frequency, high-amplitude noise at regions of nonlinearity in the velocity estimate are almost completely absent, whereas they were observed at almost every peak and trough for the position profiles. This noise-like deviation appears to have been traded-off with a bias-like offset at those same regions.

Finally, we examine the attitude estimation in terms of the quaternions, where we note that the quaternion vector at each timestep could very easily be converted into the corresponding Euler angle representation of the attitude. However, since the DCM, quaternion, and Euler angle representations are all equivalent, we elected to work with the quaternion representation since the GT attitude information is provided in that domain.

Fig. 5.5: OL TC EKF Attitude Estimate and Error.



Seeing as how the subplots on the right of Fig. 5.5 showcase the same covariance drift when no measurement update is available, we first reexamine the error subplots of Fig. 5.4 to understand why such a behavior was not as apparent with the velocity estimate as it was for the position and attitude estimates. Indeed, the pessimistic velocity covariance profiles do exhibit such a behavior once closely inspected, though with a much less severe dissentious nature. This is likely due to the fact that position and attitude estimates are much more reliant on measurement updates as the pseudo-range rates are not even necessary for the filter to execute properly.

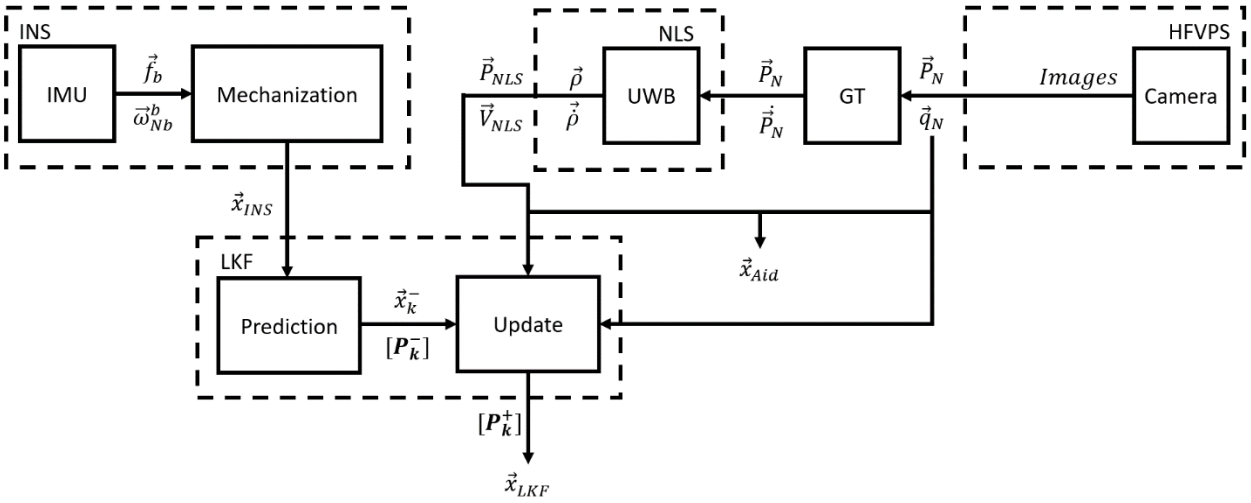
On the other hand, we can say that the attitude error is well-bonded, such that filter tuning allowed for a fair tradeoff between prediction and update reliance of the position and attitude state estimates. Moreover, the error of the attitude resembles closely the near-constant zero-valued error exhibited by the position error. However, there appear to be some error spikes in all error profiles, occurring exactly at the same time for all components. These error spikes are clearly reflected in the estimate subplots to the left of Fig. 5.5, and their synchronization between all four components indicates that the attitude quaternion is retaining its normalized status even though no explicit normalization attempt was implemented within the algorithm.

Overall, the OL TC EKF results appear to be an accurate reflection of the intended dynamics of system and how it was made to handle the various complications involved with modelling. The extensive filter tuning bore fruitful results as the position and attitude errors were both well-bounded within their respective covariances, despite the some errors in particular regions.

V.II. OL LC LKF

Next, the open-loop, loosely-coupled Kalman filter architecture will be examined. This architecture uses a linear, and hence optimal, KF to estimate the state using mechanized IMU, or INS, state to predict the state and then processed UWB measurements and camera quaternion measurements to update the state. The UWB measurements are processed using an NLS algorithm, working in error domain, to provide a standalone position and velocity state estimate.

Fig. 5.6: OL LC LKF Architecture.



According to the above-shown sensor fusion architecture from Fig. 5.6, the governing equations for an OL LC LKF are Eqns. 5.3, 5.4 & 5.5 below, where the measurement update is a 10x1 vector composed of NLS position and velocity vector estimates and four quaternion measurements. Concurrently, the relationship between the state vector '$\vec{x}$' and the measurement vector '$\vec{y}$' is fully linear and the state-to-measurement matrix '$[\boldsymbol{H}]$' is a 10x10 identity. Thus, due to the NLS estimation, it is possible to make use of an LKF, which is an optimal estimator.

e-Domain NLS

$$[\Delta\vec{P} \quad \Delta\vec{V}]^T = \left[[\boldsymbol{G}]^T[\boldsymbol{W}][\boldsymbol{G}]\right]^{-1}[\boldsymbol{G}]^T[\boldsymbol{W}]\left[\Delta\vec{\rho} \quad \Delta\dot{\vec{\rho}}\right]^T \tag{5.3}$$

$$[\vec{P} \quad \vec{V}]_{NLS}{}^T = [\Delta\vec{P} \quad \Delta\vec{V}]^T + [\vec{P} \quad \vec{V}]_0{}^T$$

Prediction

$$\vec{x}_k^- = [\boldsymbol{A}]\vec{x}_{k-1}^+ + [\boldsymbol{B}]\vec{u}_{k-1} \tag{5.4}$$

$$[\boldsymbol{P_k^-}] = [\boldsymbol{A}][\boldsymbol{P_{k-1}^+}][\boldsymbol{A}]^{-1} + [\boldsymbol{Q}]$$

Update

$$[\boldsymbol{K}] = [\boldsymbol{P_k^-}]\left[[\boldsymbol{P_k^-}] + [\boldsymbol{R}]\right]^{-1}$$

$$\vec{x}_k^+ = \vec{x}_k^- + [\boldsymbol{K}](\vec{y}_k - \vec{x}_k^-) \tag{5.5}$$

$$[\boldsymbol{P_k^+}] = \left[[\boldsymbol{I_{10}}] - [\boldsymbol{K}]\right][\boldsymbol{P_k^-}]$$

For this particular NLS, the NLS Jacobian '$[\boldsymbol{G}]$' is as follows, which is numerically determined at every iteration's initial guess for the state estimate $[\vec{P} \quad \vec{V}]_0^T$.

$$[\boldsymbol{G}] = \begin{bmatrix} \frac{x-x_1}{\rho_1} & \frac{y-y_1}{\rho_1} & \frac{z-z_1}{\rho_1} & & & \\ \vdots & \ddots & \vdots & & & \\ \frac{x-x_5}{\rho_5} & \frac{y-y_5}{\rho_5} & \frac{z-z_5}{\rho_5} & & & \\ \frac{V_x}{\rho_1} & \frac{V_y}{\rho_1} & \frac{V_z}{\rho_1} & \frac{x-x_1}{\rho_1} & \frac{y-y_1}{\rho_1} & \frac{z-z_1}{\rho_1} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{V_x}{\rho_5} & \frac{V_y}{\rho_5} & \frac{V_z}{\rho_5} & \frac{x-x_5}{\rho_5} & \frac{y-y_5}{\rho_5} & \frac{z-z_5}{\rho_5} \end{bmatrix}_{10\times 10}$$

Again, we compare the state estimate with the GT state and the error with the state covariance in Figs. 5.7, 5.8, & 5.9 below, noting that with this architecture, we have a standalone state estimate that uses only aiding sensor information. This estimate can be examined independently, but is typically neglected when less than the necessary number of measurements is available. This is because the NLS cannot be run if the necessary and sufficient number of measurements is not available, which is the case for this particular application except every 100 steps.

Fig. 5.7: OL LC LKF Position Estimate and Error.

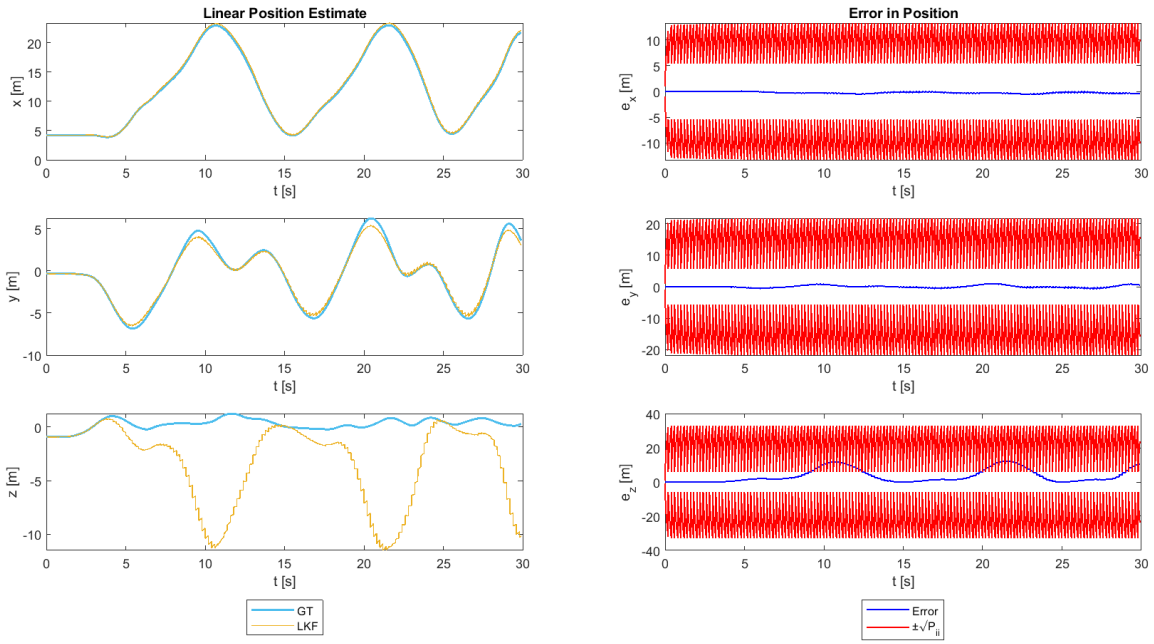
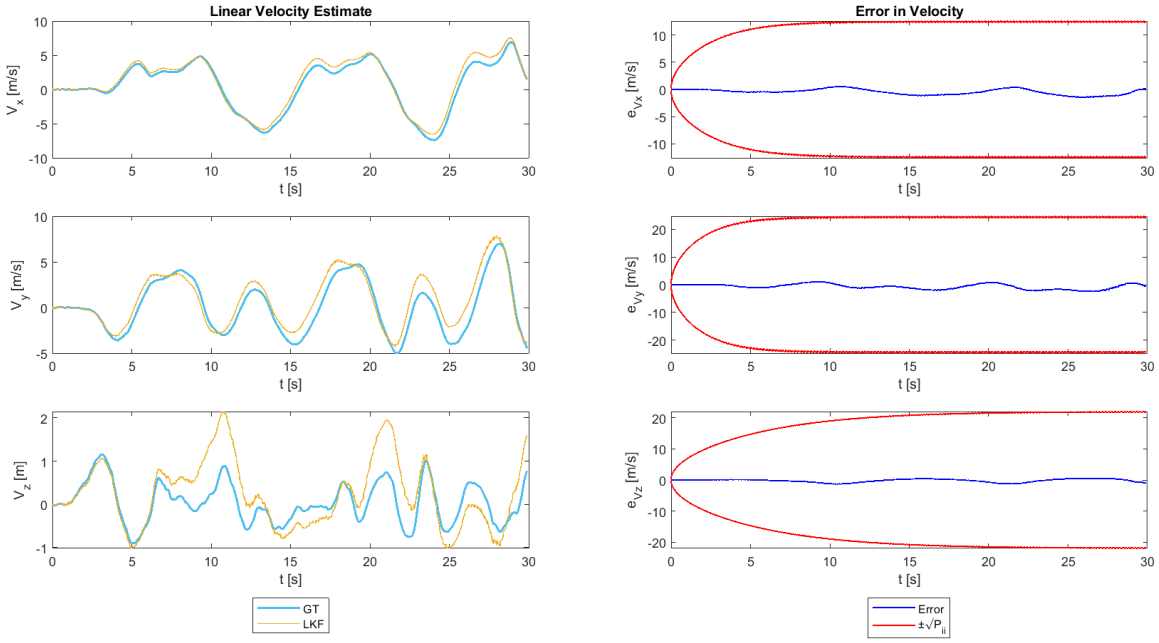


Fig. 5.7 clearly shows that the filter was not as properly tuned as it was for the OL TC EKF case, obviously because the covariance matrices were carried-on from the OL TC EKF to this OL LC LKF. Hence, although this estimate is optimal, the error is not as well-bounded within the covariance profiles, especially the z-axis error, which crosses the covariance profiles every cycle of the motion or so (i.e., every 10 [s]).

Additionally, we can see that while the TC EKF position estimates exhibited only noise-like deviation at the peaks and troughs, the LC LKF counterpart exhibited both that and a bias-like offset. Even beyond that, the z-axis estimate is clearly severely flawed, which heavily compromises the entire trajectory estimate. As mentioned prior, the z-axis estimates suffered from the GDOP effect due to the placement of the UWB transmitters. This error likely combined with the inherent suboptimality of the e-domain NLS to severely impede upon the accuracy of the z-axis position update. This is because in an LC KF, the main disadvantage, aside form the lack of updates due to the unavailability of the needed number of measurements, is that the update is only as accurate as the iteration error threshold is. Hence, there is a significant tradeoff between the number of iterations, or computational load, and the accuracy of the update, all to have a standalone aiding sensor estimate of the same state.

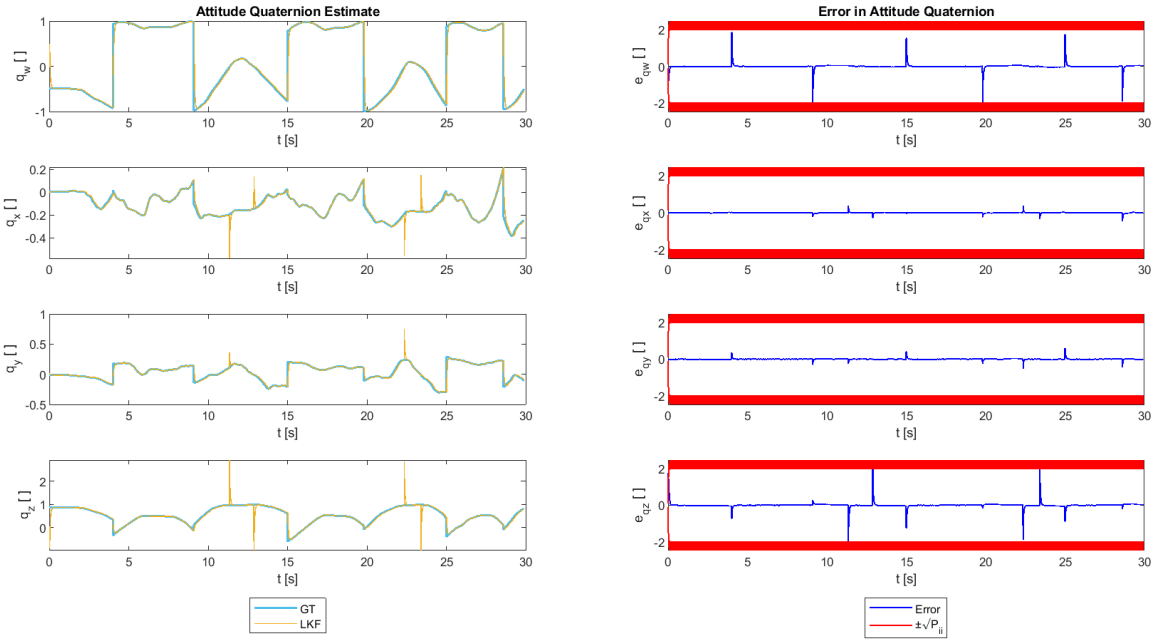
Fig. 5.8: OL LC LKF Velocity Estimate and Error.



On the other hand, the LC LKF estimates and errors for the velocity showcased in Fig. 5.8 exhibited the same overall behavior as its TC EKF counterpart, spare for a greater bias-like offset in the z-axis velocity estimate, which may again be attributed to the postulated GDOP and NLS errors compounding on one another to impede on the z-axis estimates of linear vectors.

Finally, for the attitude quaternion estimate, since the measurement update and overall filter dynamics and covariance characteristics were identical to the TC EKF, since the EKF Jacobian had a 4x4 identity mapping the HFVPS output of the quaternions onto the INS quaternion estimate, we expect the LC LKF and TC EKF to yield the same exact quaternion estimates. Thus, we may say that the attitude estimation for the TC EKF is simply an LC LKF estimate, which is an optimal integrated attitude estimation filter that also provides a standalone attitude estimate purely using aiding sensors.

Fig. 5.9: OL LC LKF Attitude Estimate and Error.



Simply contrasting the subplots of Figs. 5.5 & 5.9 indicates that our earlier conclusions regarding the matching of the attitude estimation of both the TC EKF and LC LKF hold true, meaning that the comments made about the TC EKF attitude estimates and error-covariance behaviors hold true, identically, for the LC LKF attitude estimates and error-covariance behaviors.

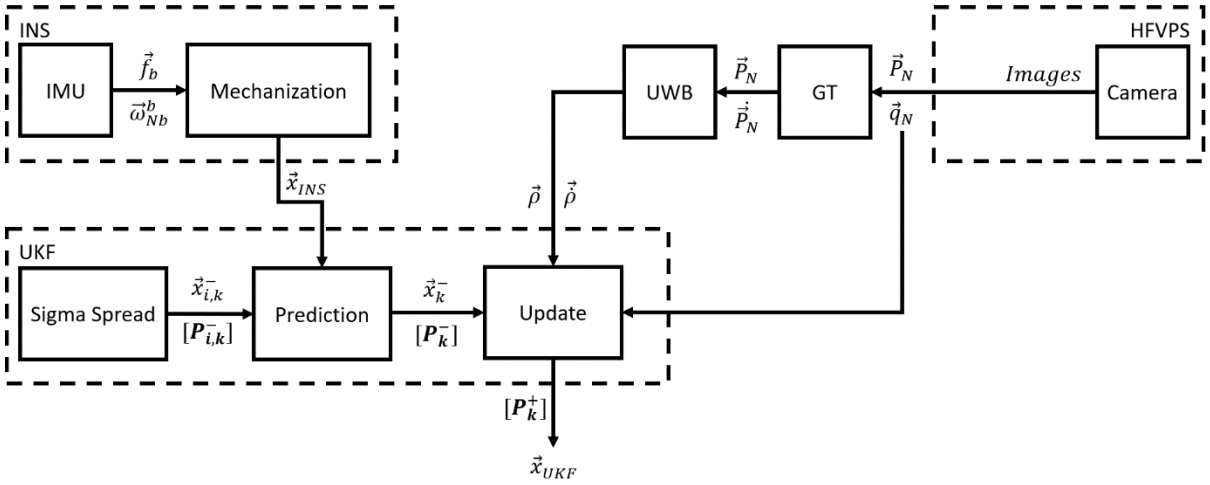
V.III. OL TC UKF

Finally, the open-loop, tightly-coupled unscented Kalman filter architecture will be examined. This architecture, akin to the OL TC EKF, uses a nonlinear, suboptimal KF to estimate the state using mechanized IMU, or INS, state to predict the state and then using raw UWB measurements and camera quaternion measurements to update the state. However, the main overarching difference is that a UKF employs an alternative method of capturing the covariance.

When examining the EKF, it was said that the reason a numerically determined Jacobian was necessary in the update block was to determine the Kalman gain and state covariance matrices. However, the innovation term, or error between the aiding measurement vector and the measurement vector simulated for the predicted state (i.e., $\vec{y}_k - h(\vec{x}_k^-)$), does not require the use of a Jacobin, though it could be used. Hence, the source of the suboptimality, or nonlinearity, in the filter is the need to use a Jacobian to compute $[\boldsymbol{K}]$ and $[\boldsymbol{P}]$.

The UKF offers an alternative method of computing both $[\boldsymbol{K}]$ and $[\boldsymbol{P}]$ by spreading sigma points, or state estimates offset from the previous state estimate, and then propagating them all within the prediction block to allow the filter a sufficiently large distribution margin of the covariance. This allows the UKF to achieve a higher-order linear approximation of a nonlinear state compared to the EKF's 1st-order TSE approximation. Thus, it is definitely superior for capturing nonlinearities.

Fig. 5.10: OL TC UKF Architecture.



According to the above-shown sensor fusion architecture from Fig. 5.10, the governing equations for an OL TC UKF are Eqns. 5.6, 5.7 & 5.8 below, where the measurement update is a 14x1 vector composed of five pseudo-range, five pseudo-range rate, and four quaternion measurements. Concurrently, by Eqns. 4.1 & 4.2, the relationship between the state vector '$\vec{x}$' and the measurement vector '$\vec{y}$' is nonlinear, spare for the attitude estimation part. Thus, it is necessary to use a nonlinear Kalman filter since the update block is nonlinear.

However, unlike for the EKF, where we required the Jacobian '$\left[\boldsymbol{H}|_{\vec{x}^-_k}\right]$', for the UKF, we use a sigma spread block. In this block, a deterministic sampling technique known as the "Unscented Transform" selects a set of sigma points around the mean, which is the previous step's state estimate. This sigma spread enables the prediction block to use standard state propagation to predict the state and state covariance of $(2n+1)$ points for an n-dimensional state vector.

$$\text{Sigma Spread } \vec{x}^+_{i,k-1} = \begin{cases} \vec{x}^+_{k-1} & ,i=0 \\ \vec{x}^+_{k-1} + \left(\sqrt{(n+\kappa)\left[\boldsymbol{P}^+_{\boldsymbol{k-1}}\right]}\right)_{(:,i)} & ,i=1,\dots,n \\ \vec{x}^+_{k-1} + \left(\sqrt{(n+\kappa)\left[\boldsymbol{P}^+_{\boldsymbol{k-1}}\right]}\right)_{(:,i)} & ,i=n+1,\dots,2n \end{cases} \tag{5.6}$$

$$\text{Prediction} \qquad \vec{x}^-_{i,k} = [\boldsymbol{A}]\vec{x}^+_{i,k-1} + [\boldsymbol{B}]\vec{u}_{k-1}$$

$$\vec{x}^-_k = \frac{1}{n+\kappa}\left(\kappa\vec{x}^-_{0,k} + \frac{1}{2}\sum_{i=1}^{2n}\vec{x}^-_{i,k}\right) \tag{5.7}$$

$$[\boldsymbol{P}^-_{\boldsymbol{k}}] = \frac{1}{n+\kappa}\left(\kappa\left[\vec{x}^-_{0,k} - \vec{x}^-_k\right]\left[\vec{x}^-_{0,k} - \vec{x}^-_k\right]^T + \frac{1}{2}\sum_{i=1}^{2n}\left[\vec{x}^-_{i,k} - \vec{x}^-_k\right]\left[\vec{x}^-_{i,k} - \vec{x}^-_k\right]^T\right)$$

Update $\vec{y}_{i,k}^{-} = h(\vec{x}_{i,k}^{-})$

$$\vec{y}_{k}^{-} = \frac{1}{n+\kappa}\left(\kappa\vec{y}_{0,k}^{-} + \frac{1}{2}\sum_{i=1}^{2n}\vec{y}_{i,k}^{-}\right)$$

$$[\boldsymbol{P_{yy}}] = \frac{1}{n+\kappa}\left(\kappa\left[\vec{y}_{0,k}^{-} - \vec{y}_{k}^{-}\right]\left[\left[\vec{y}_{0,k}^{-} - \vec{y}_{k}^{-}\right]\right]^{T} + \frac{1}{2}\sum_{i=1}^{2n}\left[\vec{y}_{i,k}^{-} - \vec{y}_{k}^{-}\right]\left[\left[\vec{y}_{i,k}^{-} - \vec{y}_{k}^{-}\right]\right]^{T}\right)$$

$$[\boldsymbol{P_{xy}}] = \frac{1}{n+\kappa}\left(\kappa\left[\vec{x}_{0,k}^{-} - \vec{y}_{k}^{-}\right]\left[\left[\vec{x}_{0,k}^{-} - \vec{y}_{k}^{-}\right]\right]^{T} + \frac{1}{2}\sum_{i=1}^{2n}\left[\vec{x}_{i,k}^{-} - \vec{x}_{k}^{-}\right]\left[\left[\vec{y}_{i,k}^{-} - \vec{y}_{k}^{-}\right]\right]^{T}\right) \quad (5.8)$$

$$[\boldsymbol{P_{vv}}] = [\boldsymbol{P_{yy}}] + [\boldsymbol{R}]$$

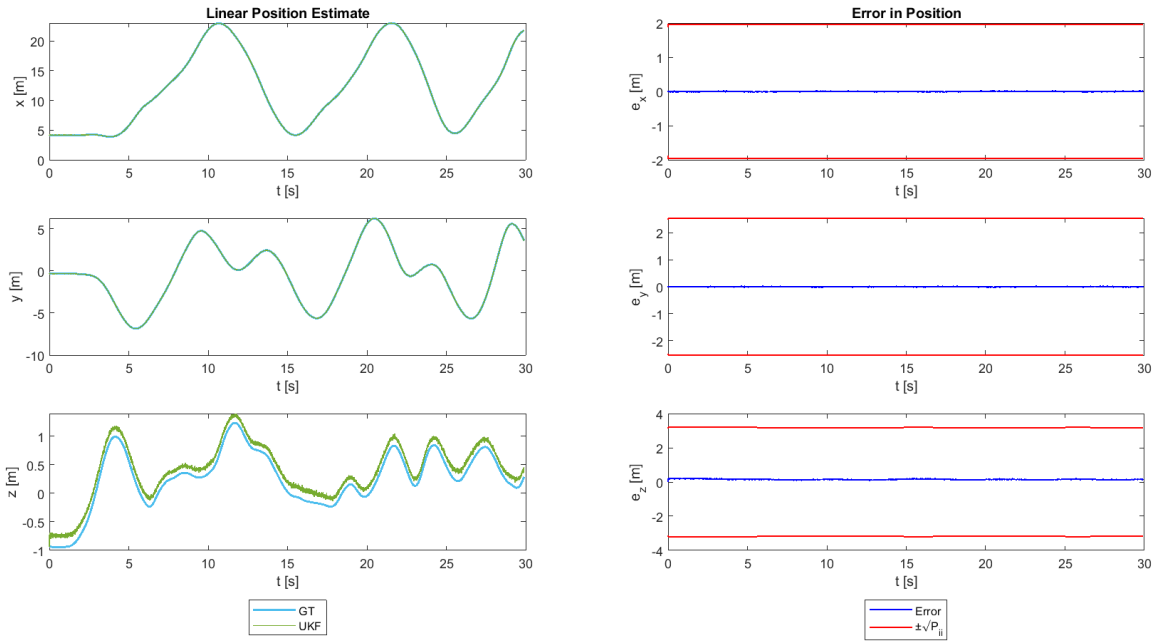
$$[\boldsymbol{K}] = [\boldsymbol{P_{xy}}][\boldsymbol{P_{vv}}]^{-1}$$

$$\vec{x}_{k}^{+} = \vec{x}_{k}^{-} + [\boldsymbol{K}](\vec{y}_{k} - \vec{y}_{k}^{-})$$

$$[\boldsymbol{P_{k}^{+}}] = [\boldsymbol{P_{k}^{-}}] - [\boldsymbol{K}][\boldsymbol{P_{vv}}][\boldsymbol{K}]^{-1}$$

In Eq. 5.6, the integer scaling parameter 'κ' is set to 0 to ensure that the mean $\vec{x}_{k-1}^{+}$ is assigned to $\vec{x}_{0,k}^{-}$, where we note that for this particular scenario, the value of 'n' is 10. Thus, the sigma spread block will output 21 different sigma points. This resulted in an extremely high computational load, so great that the UKF alone took as much time to run as both other filters combined.

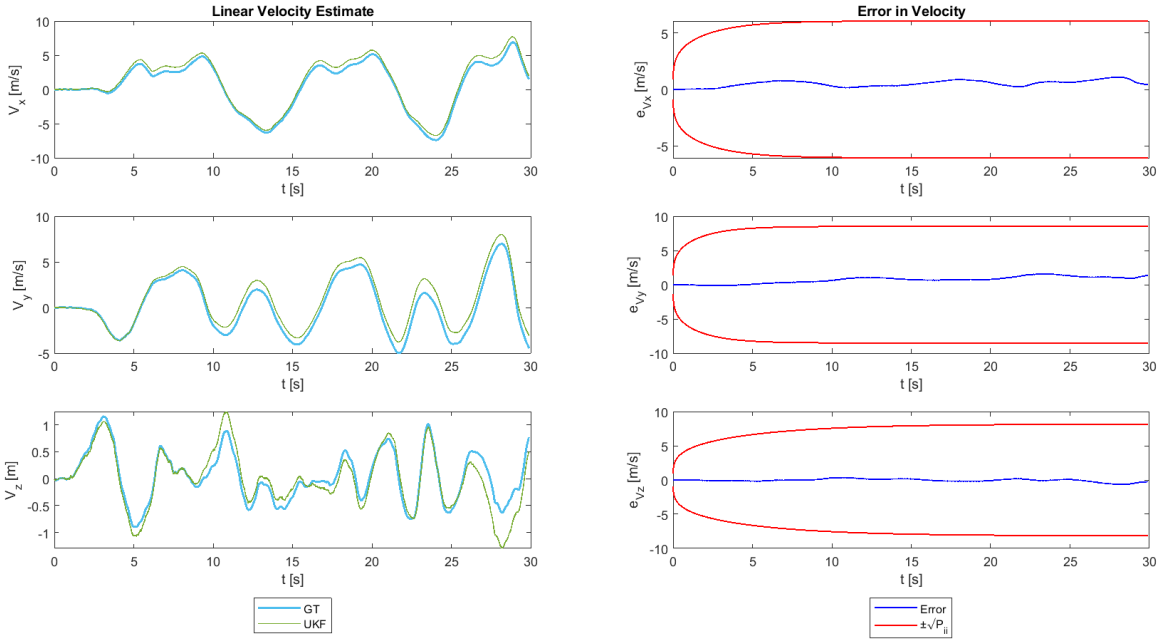
Fig. 5.11: OL TC UKF Position Estimate and Error.



An examination of Fig. 5.11 above easily shows that the OL TC UKF is undisputedly the superior filter, as its position state estimate and the GT position state are almost indistinguishable from each other, with respect to the expected and negligible noise, with a near-zero error throughout. However, a point to note is that the z-axis position estimate shows a clear and true bias-like offset at all times from the GT value. This error is at the order of magnitude of 0.1 [m], meaning that even with it, this estimate still reigns superior.

Moreover, despite the updates arriving every 100 steps for the position, the UKF proved its ability to capture the covariance with greater accuracy, such that the covariance profile drifts between the updates are almost indiscernible. However, since we used the OL TC EKF covariances, the filter appears to be slightly pessimistic, meaning that these already superb estimates could be further refined with a meticulous filter tuning process.

Fig. 5.12: OL TC UKF Velocity Estimate and Error.

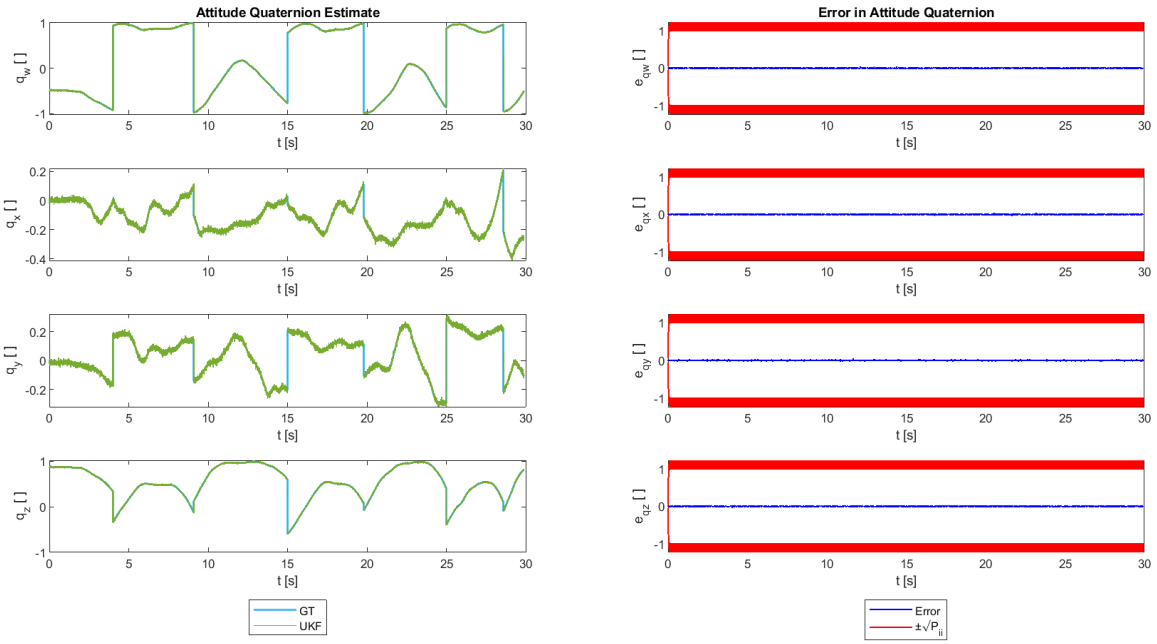


A comparison between Figs. 5.12 & 5.5 shows that the UKF was once again superior at capturing the intricacies of the highly nonlinear motion profile, as the UKF velocity estimates better aligned themselves, all axes, with the GT velocity state compared to the EKF velocity estimates. However, the issue of the bias-like offset in velocity still persists, particularly for the z-axis estimates. In fact, the z-axis UKF estimates for linear states appear to be the major, and truthfully only, source of appreciable error.

As discussed earlier, the UKF was highly sensitive to the nonlinearities of the states it estimated, meaning that the aforementioned GDOP effect on the z-axis measurements is pronounced compared to the errors in the other two axes.

Moreover, from both Figs. 5.11 & 5.12, the covariance profiles converged to a much lower steady-state value compared to the other two architectures.

Fig. 5.13: OL TC UKF Attitude Estimate and Error.



Again, from Fig. 5.13, it can be said that the UKF achieved a much more accurate estimate of the attitude, with a nearly-zero error at all times. In fact, the subplots to the right of Fig. 5.13 clearly indicates the UKF's ability to capture higher-order nonlinearities as the estimate becomes extremely noisy in highly nonlinear regions of the attitude profile. Also, we note that the steady-state values of covariance profiles were, once again, lower compared to the two other filters.

VI. Conclusion

In this paper, we outlined and implemented three open-loop Kalman filters, namely a tightly-coupled extended KF, a loosely-coupled linear KF, and a tightly-coupled unscented KF, to estimate the position, velocity, and attitude of a racing drone. All these filters were characterized with having a nonlinear measurement processing unit to integrate the IMU, UWB, and HFVPS measurements.

Executing the three filter algorithms using the covariance matrices tuned for the OL TC EKF showed that the OL LC LKF should not have been a candidate for such a highly dynamic platform in the first place as it was inferior in almost every aspect to the other two, while the OL TC UKF was clearly the superior filter to both in all regards, despite the EKF holding the advantage of being the tuned filter, spare for one: Computational load.

The results indicated that the TC EKF was the optimal trade-off between computational load and complexity and solution accuracy and reliability, as opposed to the iterative, yet linear LC KF and the robust and dynamic TC UKF. The former architecture offered a slight reduction in computational load and a standalone aiding sensor navigational solution, though at the expense of severely reduced accuracy. The latter offered greater sensitivity to the inherent nonlinearities of the motion with almost invariant zero-errors, though at the expense of an inordinately huge computational. Thus, neither alternative warranted the need to shift from the traditional OL TC EKF due to the imbalance between their perceived pros and cons with respect to the application at hand.

VII. References

[1] A. Carraro, “Application of Different Types of Kalman Filter to Approximate State of an UAV Model,” Master’s thesis, Dept. Comp. Sci., Università Ca'Foscari Venezia, Venice, Italy, 2021.

[2] K. C. Taşan, “Application of Particle Filter in The Initial Alignment Problem of Inertial Measurement Unit,” Master’s thesis, Dept. Elect. & Electron. Eng., Ankara University, Ankara, Türkiye, 2019.

[3] TDK InvenSene Inc., “MPU-9250 Product Specification,” PS-MPU-9250A-01, June 2016.

[4] The UZH FPV Dataset. (n.d.). *Dataset Files – Indoor Forward Facing – #10 – Snapdragon* [Online]. Available: https://fpv.ifi.uzh.ch/datasets/#:~:text=1.3GB-,678.7MB,-69kB