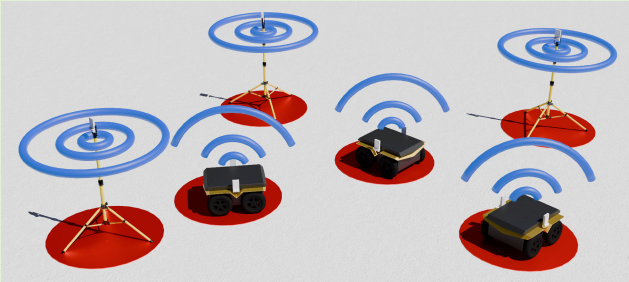
Robust Error State Sage-Husa Adaptive Kalman Filter for UWB Localization

Marius Juston, Soumil Gupta, Shrey Mathur, William R. Norris, *Member, IEEE*, Dustin Nottage, and Ahmet Soylemezoglu

***Abstract*—Given the sensors' path and interference mitigation capabilities, ultra-wideband (UWB)-based positioning systems have demonstrated high accuracy and reliability. This work aims to improve the Sage-Husa fuzzy adaptive filter (SHFAF) proposed in previous works by modifying the motion model to a 3-D ground-based differential drive robot using IMU and wheel encoder kinematic fused control inputs. In addition to the changed motion model kinematics, this article improved the positive definite constraint on P and R during dynamic estimations, thus making the filter more robust to outliers. An improvement to the computation and derivation of the fuzzy logic system for the SHFAF based on the adaptive neuro-fuzzy inference system (ANFIS) structure was developed, and training the fuzzy system using gradient descent was applied to improve the system's accuracy. Experimental validation was conducted using real-world data from a Clearpath Jackal robot equipped with Qorvo UWB sensors and static nodes. Regarding localization accuracy, the proposed velocity-based SHFAF (VelSHFAF) system outperformed the previous SHFAF implementation by approximately 30%–25% across two test courses, demonstrating its enhanced performance and reliability.**



***Index Terms*— Adaptive error state Kalman filter (KF), adaptive neuro-fuzzy inference system (ANFIS), differential drive, Sage-Husa fuzzy adaptive filter (SHFAF), stochastic gradient descent, ultra-wideband (UWB).**

I. Introduction

A. Overview

With the need for accurate localization in a noisy environment, ultra-wideband (UWB) is an ideal choice for sensor measurement due to its accurate positioning capabilities, immunity against multipath fading, and resilience against active and passive interference [1], [2]. UWB sensors utilize a large frequency spectrum from 3.1 to 10.6 GHz, with bandwidths of 500+ MHz, which help avoid multipath and interference errors common in challenging environments such as indoor situations. given the characteristics of the UWB sensors, they are ideal for a localization system in GPS-free positioning in complex environments. The need for GPS-free positioning, such as in backyards and indoor situations, is important for robotics applications such as automated mowing, indoor robotic fleet tracking, or construction site automation applications and has been a growing field of interest in new and emerging industries. UWB's ability to avoid such errors allows higher accuracy than traditional optical sensors or less powerful radars. Thus, they retain the ability to sense and communicate in obstacle-heavy terrain and allow for more robust localization systems [3].

In addition to UWB signals to transmit external measurement information for perception, robots also use internal sensors to record their states. The most common sensors for mobile ground robots are inertial measurement units (IMUs) and wheel encoders. The IMU sensors record the robot's acceleration, which is integrated over time to derive the robot's position, while the wheel encoders measure the rotation of the wheel's axes, and, given the wheels' diameter, the robot's position can also be inferred. However, both sensors come with issues of their own. IMU's most common issue is IMU drift, where the sensor integration error over more extended periods accumulates and causes the target position to diverge

Received 24 January 2025; revised 4 March 2025; accepted 5 March 2025. Date of publication 13 March 2025; date of current version 1 May 2025. This work was supported in part by the U.S. Army Corps of Engineers Engineering Research and Development Center, Construction Engineering Research Laboratory under Grant W9132T23C0013. The associate editor coordinating the review of this article and approving it for publication was Dr. Wei Tang. *(Corresponding author: Marius Juston.)*

Marius Juston, Shrey Mathur, and William R. Norris are with The Grainger College of Engineering, Industrial and Enterprise Systems Engineering Department, University of Illinois Urbana-Champaign, Urbana, IL 61801 USA (e-mail: mjuston2@illinois.edu; smath24@illinois.edu; wrnorris@illinois.edu).

Soumil Gupta is with The Grainger College of Engineering, Electrical and Computer Engineering Department, University of Illinois Urbana-Champaign, Urbana, IL 61801 USA (e-mail: soumilg2@illinois.edu).

Dustin Nottage and Ahmet Soylemezoglu are with the Construction Engineering Research Laboratory, U.S. Army Corps of Engineers Engineering Research and Development Center, Champaign, IL 61822 USA (e-mail: dustin.s.nottage@erdc.dren.mil; ahmet.soylemezoglu@erdc.dren.mil).

Digital Object Identifier 10.1109/JSEN.2025.3549315

from its actual position [4]. Odometry measurement can also introduce errors due to external issues such as wheel slip, wheel size changes, encoder accuracy, and surface irregularities surface [5].

The kinematic/measurement model for these position systems is generally nonlinear; thus, the linear Kalman filter (KF) variation has been developed to fuse this data. The main nonlinear variations of the filter are the extended Kalman filter (EKF) [6], the unscented Kalman filter (UKF) [7], the curvature Kalman filters (CKFs) [8], and the error state Kalman filter (ESKF) [9]. All of which improve the nonlinear state estimation capabilities by sampling and/or linearizing the system to approximate the probability distribution.

The main contribution of ESKF is separating the system into the nominal and the error state (coming from the IMU drift and wheel encoder inaccuracies). The changes in the error dynamics are slow (IMU drift happens slowly over time) because all the large-signal dynamics have been integrated into the nominal state. Given the lower dynamic speed of the error state, the Kalman filter measurement corrections can be applied at a lower rate than the predictions, which can help reduce computational requirements [10].

However, the issue with the measurement process model is the assumption that the noise is time-invariant. Thus, their performance degrades when the assumption no longer holds. Adaptive Kalman filters are used to help solve this issue. The simplified Sage-Husa adaptive filter (SHAF) has been used to estimate the time-variant noise covariances [11], [12], [13]. The SHAF is an adaptive version of the ESKF in the viewpoint that the constant measurement noise covariance matrices, R, and the process noise covariance matrices, Q, are estimated adaptively given the input signals of the KF, as such complex time-varying noise from changing environmental factors can be estimated dynamically [14], thus improving the reliability of the Kalman filter.

Even with the adaptive noise estimation, outliers still significantly affect adaptive Kalman filters [15]. This is because the adaptive noise estimation assumes that there is no considerable variation of the measurement distribution within a small period, as such techniques such as employing fuzzy logic systems [16], [17], [18] and chi-square tests have been developed [19] to detect such measurement points and reweighing techniques have been employed to correct the outlier, allowing them to still contribute to the state estimation [20], [21], while keeping the state estimation robust.

However, many of the algorithms or implementations rely solely on IMU and UWB measurements [16], [22], [23] in a 2-D context, which is what our article proposes to change and improve. By relying instead on the robot's linear and angular velocities, we get better accuracy while implementing a quaternion-based 3-D kinematic system while also relying and improving the SHFAF's outlier filtering algorithm to better handle outlier inputs.

B. Contribution

This article provides several improvements to the Sage-Husa fuzzy adaptive filter (SHFAF) paper [16] for UWB-ranging applications while performing adaptive outlier detection using a fuzzy inference system and measurement reweighting to correct the data points.

1) The model dynamics leverage fused wheel and IMU odometry information, utilizing velocity and angular velocity estimates as control inputs rather than relying on integrated IMU measurements or accelerations.
2) The 3-D quaternion dynamics model for differential drive systems is derived, including the error state model and closed-form solutions for error state dynamics. Additionally, bounds on hyperparameters for the covariance update (e.g., the R_k matrix) are derived, ensuring system stability and robustness
3) A computationally optimized fuzzy logic inference system is implemented, extending prior adaptive neuro-fuzzy inference system (ANFIS) architectures and enabling differential fuzzy logic system training. The implementation includes trainable smooth saturation functions within the ANFIS structure for enhanced output smoothness and performance optimization.
4) The system's performance is demonstrated with real-world data on a Clearpath Jackal robot equipped with static nodes and Qorvo UWB sensors.

Sections II–VI derive the velocity-based error state Kalman filter model based on [10], with the angular and linear velocity as the control inputs. Section II derives 3-D quaternion dynamics and error state model based on velocity. Following the derivation of the dynamics of the model, the improvements to the SHFAF are depicted by improving the stability of the R update, the constraints of averaging constant $s_k^{\alpha} d_k$ and minimum k_s for the system's stability. Sections VII and VIII goes through the fuzzy logic system and optimization of the model based on the ANFIS structure derived in [24]. Finally, experimentation results are illustrated by comparing SHFAF with the new velocity-based SHFAF (VelSHFAF) and VelSHFAF with square root outlier reweighting (VelSHFAFSqrtM) on two test courses. Additionally, a test of training the ANFIS structure is presented.

II. Velocity-Based SHFAF

A. True-State Kinematics

The robot followed a differential drive model in 3-D space; the derivation of the SHFAF, like most Kalman filter models, started with the derivation of the true state kinematics. However, the SHFAF followed an error state Kalman filter model

$$\dot{p}_t = q_t \otimes \begin{bmatrix} v_t & 0 & 0 \end{bmatrix}^{\mathrm{T}} \otimes q_t^* = R_t\{q\} \begin{bmatrix} v_t & 0 & 0 \end{bmatrix}^{\mathrm{T}} \quad (1)$$

$$\dot{q}_t = \frac{1}{2} q_t \otimes \omega_t \quad (2)$$

$$\dot{\omega}_{bt} = \omega_w \quad (3)$$

$$\dot{v}_{bt} = v_w \quad (4)$$

where $R = R\{q\} = (q_w^2 - q_v^{\top} q_v) I + 2 q_v q_v^{\top} + 2\, q_w [q_v]_{\times}$ defined as the Rodrigues' rotation formula. Here, $q_t = [q_w \; q_x \; q_y \; q_z]$ is the quaternion, where q_w is the real part and $[\cdot]_{\times}$ the skew-symmetric matrix. In this dynamical system, the control input for the true velocity v_t, the forward linear velocity, and the 3-D angular rate ω_t were derived from the

noisy sensor data model from the fusion of wheel encoders and IMU. The measurements v_m and ω_m represented the noisy sensor readings in the body frame, and v_{bt} and ω_{bt} were bias terms

$$\omega_m = \omega_t + \omega_{bt} + \omega_n \tag{5}$$
$$v_m = v_t + v_{bt} + v_n. \tag{6}$$

With this, the true values can be isolated

$$\omega_t = \omega_m - \omega_{bt} - \omega_n \tag{7}$$
$$v_t = v_m - v_{bt} - v_n. \tag{8}$$

Substituting to the previous kinematics equation

$$\dot{p}_t = R_t \begin{bmatrix} (v_m - v_{bt} - v_n) & 0 & 0 \end{bmatrix}^{\mathrm{T}} \tag{9}$$
$$\dot{q}_t = \frac{1}{2} q_t \otimes (\omega_m - \omega_{bt} - \omega_n) \tag{10}$$
$$\dot{\omega}_{bt} = \omega_w \tag{11}$$
$$\dot{v}_{bt} = v_w \tag{12}$$

named as $\dot{x}_t = f_t(x_t, u, w)$. This system had states x_t, controlled by the odometry noisy readings u_m, and was perturbed by Gaussian noise w, all defined by

$$x_t = \begin{bmatrix} p_t \\ q_t \\ \omega_{bt} \\ v_{bt} \end{bmatrix}, \quad u = \begin{bmatrix} \omega_m - \omega_n \\ v_m - v_n \end{bmatrix}, \quad w = \begin{bmatrix} \omega_w \\ v_w \end{bmatrix}. \tag{13}$$

As the SHFAF is an error state Kalman filter, the dynamics were separated into two kinematic systems: the nominal and the error states.

B. Nominal-State Kinematics

The nominal-state kinematics corresponded to the modeled system without noises or perturbations (the noise perturbation $w = 0$) [10]

$$\vec{v} = \begin{bmatrix} v_m - v_b & 0 & 0 \end{bmatrix}^{\mathrm{T}} \tag{14}$$
$$\dot{p} = R\vec{v} \tag{15}$$
$$\dot{q} = \frac{1}{2} q \otimes (\omega_m - \omega_b) \tag{16}$$
$$\dot{\omega}_b = 0 \tag{17}$$
$$\dot{v}_b = 0. \tag{18}$$

In discrete time, the nominal state is expressed as follows:

$$p \leftarrow p + R\vec{v}\Delta t \tag{19}$$
$$q \leftarrow q \otimes q\{(\omega_m - \omega_b)\Delta t\} \tag{20}$$
$$\omega_b \leftarrow \omega_b \tag{21}$$
$$v_b \leftarrow v_b \tag{22}$$

where $x \leftarrow f(x, \cdot)$ stands for a time update of the type $x_{k+1} = f(x_k, \cdot_k)$, and $q\{w\} = e^{\phi u/2} = \begin{bmatrix} \cos(\phi/2) \\ u \sin(\phi/2) \end{bmatrix}$. The system was evaluated using standard Euler integration. More advanced computing-heavy integration methods, such as Range-Kutta (RK4), could produce more accurate results.

C. Error-State Kinematics

The goal was to determine the linearized dynamics of the error state. Each state equation's composition was provided and solved for the error state, and all second-order terms were removed. These second-order error terms do not significantly impact the system and thus could be removed without concern for accuracy. The full-error-state dynamic system was demonstrated here, and the proofs were derived in the following subsections. The skew-symmetric operator is denoted and defined as $[\cdot]_\times$

$$[\cdot]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \tag{23}$$
$$v_x = \begin{bmatrix} \begin{bmatrix} (v_m - v_b) & 0 & 0 \end{bmatrix}^{\mathrm{T}} \end{bmatrix}_\times \tag{24}$$
$$v_x = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -(v_m - v_b) \\ 0 & v_m - v_b & 0 \end{bmatrix} \tag{25}$$
$$\dot{\delta p} = -Rv_x\delta\theta - R\delta v_b - Rv_n \tag{26}$$
$$\dot{\delta\theta} = -[\omega_m - \omega_b]_\times\delta\theta - \delta\omega_b - \omega_n \tag{27}$$
$$\dot{\delta\omega_b} = \omega_w \tag{28}$$
$$\dot{\delta v_b} = v_w. \tag{29}$$

1) Linear Position Error: The goal was to determine $\dot{\delta p}$, which represented the dynamics of the position errors, starting from the following relations:

$$R_t = R(I + [\delta\theta]_\times) + O(\|\delta\theta\|^2) \tag{30}$$
$$\dot{p} = R \begin{bmatrix} v_\mathcal{B} & 0 & 0 \end{bmatrix}^{\mathrm{T}} \tag{31}$$

where $(I + [\delta\theta]_\times)$ was the local perturbation contribution from the error term, approximated using Taylor series rotation expansion [10]. The terms $v_\mathcal{B}$ and $\delta v_\mathcal{B}$ were defined as the body frame's large- and small-signal velocities

$$v_\mathcal{B} = v_m - v_b \tag{32}$$
$$\delta v_\mathcal{B} = -\delta v_b - v_n. \tag{33}$$

The actual velocity could be written in the inertial frame as a composition of large- and small-signal terms

$$v_t = v_\mathcal{B} + \delta v_\mathcal{B}. \tag{34}$$

The term $\dot{p}_t$ is expressed in two different forms (left and right developments), where the terms $O(\|\delta\theta\|^2)$ had been ignored.

Annotating $\vec{v}_\mathcal{B} = [v_t \ 0 \ 0]^{\mathrm{T}}$ and $\vec{\delta v}_\mathcal{B} = [\delta v_t \ 0 \ 0]^{\mathrm{T}}$

$$\dot{p} + \dot{\delta p} = \dot{p}_t = R_t v_t \tag{35}$$
$$\dot{p} + \dot{\delta p} = R_t(\vec{v}_\mathcal{B} + \vec{\delta v}_\mathcal{B}) \tag{36}$$
$$R\vec{v}_\mathcal{B} + \dot{\delta p} = R(I + [\delta\theta]_\times)(\vec{v}_\mathcal{B} + \vec{\delta v}_\mathcal{B}) \tag{37}$$
$$R\vec{v}_\mathcal{B} + \dot{\delta p} = R\vec{v}_\mathcal{B} + R\vec{\delta v}_\mathcal{B} + R[\delta\theta]_\times\vec{v}_\mathcal{B} + R[\delta\theta]_\times\vec{\delta v}_\mathcal{B}. \tag{38}$$

After removing $R\vec{v}_\mathcal{B}$ from both sides and disregarding the term $R[\delta\theta]_\times\vec{\delta v}_\mathcal{B}$ given that it was a second-order infinitesimal, the system becomes

$$\dot{\delta p} = R\vec{\delta v}_\mathcal{B} + R[\delta\theta]_\times\vec{v}_\mathcal{B} \tag{39}$$
$$\dot{\delta p} = R\left(\vec{\delta v}_\mathcal{B} + [\delta\theta]_\times\vec{v}_\mathcal{B}\right). \tag{40}$$

Reorganizing some cross-products (with $[a]_\times b = -[b]_\times a$), the equation became

$$\delta\dot{p} = R\left(\delta\vec{v}_{\mathcal{B}} - [\vec{v}_{\mathcal{B}}]_\times \delta\theta\right) \tag{41}$$

then recalling, with $\delta\vec{v}_b = [\delta v_b \ 0 \ 0]^{\mathrm{T}}$ and that $\vec{v_n} = [v_n \ 0 \ 0]^{\mathrm{T}}$

$$\delta\dot{p} = R\left(-\delta\vec{v}_b - \vec{v_n} - v_x\delta\theta\right) \tag{42}$$

$$\delta\dot{p} = -Rv_x\delta\theta - R\delta\vec{v}_b - R\vec{v_n}. \tag{43}$$

2) Orientation Error: The linear orientation error was derived similarly to the linear position error and could be demonstrated in [10, Eq. 256-259] for deriving the error dynamics for (27).

D. Error State Closed-Form Integration Method

The error state dynamics were posed as a linear system, and the closed-form solution of the system could thus be derived.

Consider the full velocity-odometry system, which could be posed as

$$\delta\dot{x} = A\delta x + Bw \tag{44}$$

where

$$\delta\dot{p} = -Rv_x\delta\theta - R\delta v_b - Rv_n \tag{45}$$

$$\delta\dot{\theta} = -[\omega_m - \omega_b]_\times\delta\theta - \delta\omega_b - \omega_n \tag{46}$$

$$\delta\dot{\omega}_b = \omega_w \tag{47}$$

$$\delta\dot{v}_b = v_w. \tag{48}$$

For which $-R\delta v_b \rightarrow -R[\delta v_b \ 0 \ 0]^{\mathrm{T}} = -R[1 \ 0 \ 0]^{\mathrm{T}}\delta v_b = -RU\delta v_b$. Thus

$$\delta\dot{x} = \begin{bmatrix}\delta\dot{p} & \delta\dot{\theta} & \delta\dot{\omega}_b & \delta\dot{v}_b\end{bmatrix}^{\mathrm{T}}. \tag{49}$$

Whose discrete time integration required the computation of the following transition matrix:

$$\Phi = \sum_{k=0}^{\infty} = \frac{1}{k!}A^k\Delta t^k = I + A\Delta t + \frac{1}{2}A^2\Delta t^2 + \cdots. \tag{50}$$

A closed form of the transition matrix Φ was desired. The dynamics matrix A could be determined by examining the original equations

$$P_\theta = -Rv_x \tag{51}$$

$$P_v = -RU \tag{52}$$

$$\Theta_\theta = -[\omega_m - \omega_b]_\times \tag{53}$$

$$\Theta_\omega = -I. \tag{54}$$

Thus, A became

$$A = \begin{bmatrix}0 & P_\theta & 0 & P_v\\ 0 & \Theta_\theta & \Theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}. \tag{55}$$

Its integration with step time Δt is $x_{n+1} = e^{(A\Delta t)}x_n = \Phi x_n$. The transition matrix Φ followed a Taylor development in increasing $A\Delta t$ powers. The following powers of A are derived to identify the trends of Φx_n for $k > 1$ [10]:

$$A = \begin{bmatrix}0 & P_\theta & 0 & P_v\\ 0 & \Theta_\theta & \Theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$$

$$A^2 = \begin{bmatrix}0 & \Theta_\theta P_\theta & P_\theta\theta_\omega & 0\\ 0 & \Theta_\theta^2 & \Theta_\theta\theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix} \tag{56}$$

$$A^3 = \begin{bmatrix}0 & \Theta_\theta^2 P_\theta & \Theta_\theta P_\theta\theta_\omega & 0\\ 0 & \Theta_\theta^3 & \Theta_\theta^2\theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}$$

$$A^4 = \begin{bmatrix}0 & \Theta_\theta^3 P_\theta & \Theta_\theta^2 P_\theta\theta_\omega & 0\\ 0 & \Theta_\theta^4 & \Theta_\theta^3\theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix} \tag{57}$$

from which it is now visible that, for $k > 1$,

$$A^k = \begin{bmatrix}0 & \Theta_\theta^{k-1} P_\theta & \Theta_\theta^{k-2} P_\theta\theta_\omega & 0\\ 0 & \Theta_\theta^k & \Theta_\theta^{k-1}\theta_\omega & 0\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}. \tag{58}$$

From the increasing powers of A, it could be noted that an increasing power of Θ_θ and a fixed matrix component are present in each of the blocks in the powers of A. The trend of these powers could lead to closed-form solutions. The matrix Φ could thus be segmented as follows:

$$\Phi = \begin{bmatrix}I & \Phi_{P\theta} & \Phi_{P\omega} & P_v\Delta t\\ 0 & \Phi_{\Theta\theta} & \Phi_{\Theta\omega} & 0\\ 0 & 0 & I & 0\\ 0 & 0 & 0 & I\end{bmatrix}. \tag{59}$$

1) Rotational Diagonal: Starting with the term $\Phi_{\Theta\theta}$, which relates the new angular error to the old angular error, writing the Taylor series discrete integration for this term led to

$$\Phi_{\Theta\theta} = \sum_{k=0}^{\infty}\frac{1}{k!}\Theta_\theta^k\Delta t^k = \sum_{k=0}^{\infty}\frac{1}{k!}[-(\omega_m - \omega_b)]_\times^k\Delta t^k. \tag{60}$$

which represents the following rotation matrix:

$$\Phi_{\Theta\theta} = R\{(\omega_m - \omega_b)\Delta t\}^T. \tag{61}$$

The solution corresponded to a rotation matrix in the form $\Phi = R\{-u\Delta\theta\} = R\{\omega\Delta t\}^T$, according to the Rodrigues' rotation formula ($R\{u\Delta\theta\} = I + \sin\Delta\theta[u]_\times + (1 - \cos\Delta\theta)[u]_\times^2$).

2) Position Versus Angle Term: The term $\Phi_{P\theta}$ could be written as its Taylor series

$$\Phi_{P\theta} = P_\theta\Delta t + \frac{1}{2}P_\theta\Theta_\theta\Delta t^2 + \frac{1}{3!}P_\theta\Theta_\theta^2\Delta t^3 + \cdots \tag{62}$$

$$= P_\theta\Delta t\left(I + \frac{1}{2}\Theta_\theta\Delta t + \frac{1}{3!}\Theta_\theta^2\Delta t^2 + \cdots\right) \tag{63}$$

$$= P_\theta\Delta t\left(I + \sum_{k\geq 1}\frac{(\Theta_\theta\Delta t)^k}{(k+1)!}\right). \tag{64}$$

Factoring P_θ and writing

$$\Phi_{P\theta} = P_\theta\Sigma_1 \tag{65}$$

$$\Sigma_1 = I\Delta t + \frac{1}{2}\Theta_\theta\Delta t^2 + \frac{1}{3!}\Theta_\theta^2\Delta t^3 + \cdots. \tag{66}$$

The series Σ_1 resembled the series that was written for $\Phi_{\Theta\theta}$ with two exceptions.

1) The powers of Θ_θ do not match with the rational coefficients $(1/k!)$ and with the powers of Δt. The sub-index "1" in Σ_1 denotes the fact that one power of Θ_θ is missing in each of the members.
2) Some terms at the start of the series are missing, thus the "1."

The first issue could be solved by applying the equality for $[\cdot]_\times^k$ as

$$[u]_\times^2 = uu^T - I, \quad [u]_\times^3 = -[u]_\times, \ [u]_\times^4 = -[u]_\times^2, \dots \tag{67}$$

The second issue was solved by applying $[u]_\times^3 = -[u]_\times$ to Θ_θ [10]

$$\Theta_\theta = -[\omega_m - \omega_b]_\times = \frac{[\omega_m - \omega_b]_\times^3}{\|\omega_m - \omega_b\|^2} = \frac{-\Theta_\theta^3}{\|\omega_m - \omega_b\|^2} = \frac{-\Theta_\theta^3}{\|\omega\|^2}. \tag{68}$$

This allowed an increase in the components of Θ_θ in the series by two, and writing it as if $\omega \neq 0$

$$\Sigma_1 = I\Delta t - \frac{\Theta_\theta}{\|\omega\|^2}\left(\frac{1}{2}\Theta_\theta^2\Delta t^2 + \frac{1}{3!}\Theta_\theta^3\Delta t^3 + \cdots\right) \tag{69}$$

and $\Sigma_1 = I\Delta t$, otherwise. All the powers in the new series matched the correct coefficient, but the terms I and $\Theta_\theta\Delta t$ were missing. This could be corrected by adding and subtracting terms and substituting the full series with its closed form

$$\Sigma_1 = I\Delta t - \frac{\Theta_\theta}{\|\omega\|^2}\left(I + \Theta_\theta\Delta t + \frac{1}{2}\Theta_\theta^2\Delta t^2 + \frac{1}{3!}\Theta_\theta^2\Delta t^3 + \cdots - I - \Theta_\theta\Delta t\right) \tag{70}$$

$$= I\Delta t - \frac{\Theta_\theta}{\|\omega\|^2}\left(R\{\omega\Delta t\}^T - I - \Theta_\theta\Delta t\right). \tag{71}$$

which was the valid closed-form solution if $w \neq 0$. Therefore,

$$\Phi_{P\theta} = \begin{cases} -Rv_x\Delta t I, & \omega \to 0 \\ -Rv_x\left(I\Delta t + \dfrac{[\omega]_\times}{\|\omega\|^2}\left(R\{\omega\Delta t\}^T - I + [\omega]_\times\Delta t\right)\right). & \\ \omega \neq 0 & \end{cases} \tag{72}$$

3) Position Versus Angular Velocity Error: Moving to the term $\Phi_{P\omega}$ by writing its series

$$\Phi_{P\omega} = \frac{1}{2}P_\theta\Theta_\omega\Delta t^2 + \frac{1}{3!}\Theta_\theta P_\theta\Theta_\omega\Delta t^3 + \frac{1}{4!}\Theta_\theta^2 P_\theta\Theta_\omega\Delta t^4 + \cdots. \tag{73}$$

Similar to the $\Phi_{\Theta\theta}$ proof, factor the $\Phi_{P\omega} = (1/2)P_\theta\Theta_\omega\Delta t^2\Sigma_0$, such that

$$\Phi_{P\omega} = \frac{1}{2}P_\theta\Theta_\omega\Delta t^2\left(I + \Theta_\theta\Delta t + \frac{1}{2}\Theta_\theta\Delta t^2 + \cdots\right) \tag{74}$$

$$= \frac{1}{2}P_\theta\Theta_\omega\Delta t^2 R\{w\Delta t\}^T \tag{75}$$

$$= \frac{1}{2}(-Rv_x)(-I)\Delta t^2 R\{w\Delta t\}^T \tag{76}$$

$$= \frac{1}{2}Rv_x R\{w\Delta t\}^T\Delta t^2. \tag{77}$$

4) Angle Versus Angular Velocity Error: The term $\Phi_{\Theta\omega}$ was provided as a series [10] (387)

$$\Phi_{\Theta\omega} = \Theta_\omega\Delta t + \frac{1}{2}\Theta_\theta\Theta_\omega\Delta t^2 + \frac{1}{3!}\Theta_\theta^2\Theta_\omega\Delta t^3 + \cdots. \tag{78}$$

Similar to the previous proofs, factor the $\Phi_{\Theta\omega} = \Theta_\omega\Sigma_1$, such that

$$\Phi_{\Theta\omega} = \Theta_\omega\left(I\Delta t + \Theta_\theta\Delta t + \frac{1}{2}\Theta_\theta\Delta t^2 + \cdots\right). \tag{79}$$

This formulation was exactly the same as that of $\Phi_{\Theta\theta}$, as such

$$\Phi_{\Theta\omega} = \begin{cases} -I\Delta t, & \omega \to 0 \\ -I\Delta t - \dfrac{[\omega]_\times}{\|\omega\|^2}\left(R\{\omega\Delta t\}^T - I + [\omega]_\times\Delta t\right) & \\ \omega \neq 0. & \end{cases} \tag{80}$$

5) Full-Transition Matrix Φ: The full system Φ was thus represented as following (remember that $\omega = \omega_m - \omega_b$).

If $w \to 0$

$$\Phi = \begin{bmatrix} I & -Rv_x I\Delta t & \frac{1}{2}Rv_x R\{w\Delta t\}^T\Delta t^2 & -RU\Delta t \\ 0 & R\{\omega\delta t\}^T & -I\Delta t & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix}. \tag{81}$$

When the system's $\omega \neq 0$, the transition matrix was defined in (82), as shown at the bottom of the next page.

E. Impulse Noises

The system would also include random noise applied to the position, orientation, and bias estimates, modeled by Gaussian random variables, $I_i \sim \mathcal{N}(0, \Sigma)$. The covariance matrices were obtained by integrating the covariances of $\omega_n, v_n, \omega_w, v_w$ over time-step Δt [10].

Such that

$$\delta p = \delta p_{\text{new}} + p_i \tag{83}$$

$$\delta\theta = \delta\theta_{\text{new}} + \theta_i \tag{84}$$

$$\delta\omega_b = \delta\omega_{b,\text{new}} + \omega_i \tag{85}$$

$$\delta v_b = \delta v_{b,\text{new}} + v_i. \tag{86}$$

Thus,

$$P_i = \sigma_{v_n}^2\Delta t^2 I \tag{87}$$

$$\Theta_i = \sigma_{w_n}^2\Delta t^2 I \tag{88}$$

$$\Omega_i = \sigma_{w_w}^2\Delta t I \tag{89}$$

$$V_i = \sigma_{v_w}^2\Delta t I \tag{90}$$

where $\sigma_{v_n}, \sigma_{w_n}, \sigma_{w_w}$, and σ_{v_w} were to be determined from the information in the data sheets or experiment measurements. The error state system was currently defined as follows [10]:

$$\delta x \leftarrow f(x, \delta x, u_m, i) = F_x(x, u_m)\delta x + F_i. \tag{91}$$

The SHFAF prediction was written as follows:

$$\hat{\delta x} \leftarrow F_x(x, u_m)\hat{\delta x} \tag{92}$$

$$P \leftarrow F_x P F_x^T + F_i Q F_i^T \tag{93}$$

where $\delta x \sim \mathcal{N}\{\hat{\delta x}, P\}$, F_x and F_i were the Jacobians of $f(\cdot)$ with respect to the error and perturbation vectors, respectively, and Q_i were the covariance matrix of the perturbation impulses. In this case, $F_x = \Phi$ from Section II-D5

$$F_i = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix}, \quad Q_i = \begin{bmatrix} P_i & 0 & 0 & 0 \\ 0 & \Theta_i & 0 & 0 \\ 0 & 0 & \Omega_i & 0 \\ 0 & 0 & 0 & V_i \end{bmatrix}. \tag{94}$$

The linear equation $\hat{\delta x} \leftarrow \cdots$ always returned zero, as the mean of the error δx was initialized to zero.

III. UWB Measurement Update

Suppose a sensor delivers information that depended on the state, such as [10]

$$y = h(x_t) + v \tag{95}$$

where $h(\cdot)$ was a general nonlinear function of the system state (the actual state), and v was white Gaussian noise with covariance V, $v \sim \mathcal{N}\{0, V\}$. The filter correction equations were thus

$$K = PH^T(HPH^T + V)^{-1} \tag{96}$$

$$\hat{\delta x} \leftarrow K(y - h(\hat{x_t})) \tag{97}$$

$$P \leftarrow (I - KH)P(I - KH)^T + KRK^T. \tag{98}$$

Here, P used the more numerically stable definition, which enforces the system as symmetric and positive, as the Joseph form. As noted, the system required the Jacobian matrix H to be defined with respect to the error state δx and evaluated at the true-state estimate $\hat{x_t} = x \oplus \hat{\delta x}$. As the error state mean was zero at this stage (it has not been observed yet), the system reduced to $\hat{x_t} = x$, and the nominal error x could be used as the evaluation point [10]. Leading to

$$H \equiv \left.\frac{\partial h}{\partial \delta x}\right|_x. \tag{99}$$

The Jacobian was equivalent to

$$H = H \equiv \left.\frac{\partial h}{\partial \delta x}\right|_x = \left.\frac{\partial h}{\partial x_t}\right|_x \left.\frac{\partial x_t}{\partial \delta x}\right|_x = H_x X_{\delta x}. \tag{100}$$

Here, H_x was the standard Jacobian of $h(\cdot)$ with respect to its argument. The first part of the chain rule in [(100)] depended on the measurement function of the particular sensor. The second part was based only on the error dynamics and actual state [10].

A. H_x Derivation

1) Position Only: Given that the application that the system makes was triangulated UWB system measurements. For the UWB, the output of the system resulted in an output matrix $h = [p_t]$.

As such, the output of H_x was

$$H_x = \begin{bmatrix} \frac{\partial p_t}{\partial p_t} & 0 \end{bmatrix} = \begin{bmatrix} I_3 & 0 \end{bmatrix}. \tag{101}$$

2) Range Only: Given that the UWB were just range measurements, a linearized version of H for just the range measurements could also be derived.

The measurement function was the range function: $h(x) = ((p_x - a_x)^2 + (p_y - a_y)^2 + (p_z - a_z)^2)^{1/2}$, where the equivalent H matrix was thus

$$h_{p_x} = \frac{p_x - a_x}{\sqrt{(p_x - a_x)^2 + (p_y - a_y)^2 + (p_z - a_z)^2}} \tag{102}$$

$$h_{p_y} = \frac{p_y - a_x}{\sqrt{(p_x - a_x)^2 + (p_y - a_y)^2 + (p_z - a_z)^2}} \tag{103}$$

$$h_{p_z} = \frac{p_z - a_x}{\sqrt{(p_x - a_x)^2 + (p_y - a_y)^2 + (p_z - a_z)^2}} \tag{104}$$

$$H_x = \begin{bmatrix} h_{p_x} & h_{p_y} & h_{p_z} & 0 \end{bmatrix}. \tag{105}$$

3) Offset UWB Position Only: When dealing with an offset UWB sensor on the robot, the H_x incorporated information from the heading and the position. The measurement $h(x) = p_t + q_t \otimes \mathrm{d} \otimes q_t*$, where d was the offset of the UWB sensor from the center of the robot. The linearized version of H_x was thus

$$H_x = \begin{bmatrix} \frac{\partial h(x)}{\partial p_t} & \frac{\partial h(x)}{\partial q_t} & \frac{\partial h(x)}{\partial \omega_{b_t}} & \frac{\partial h(x)}{\partial v_{b_t}} \end{bmatrix} \tag{106}$$

$$= \begin{bmatrix} I_3 & \frac{\partial h(x)}{\partial q_t} & 0 & 0 \end{bmatrix} \tag{107}$$

$$\frac{\partial h(x)}{\partial q_t} = \frac{\partial (q_t \otimes \mathrm{d} \otimes q_t*)}{\partial q_t} = \frac{\partial d_{\mathrm{rot}}}{\partial q_t}. \tag{108}$$

Furthermore, q_t could be rewritten as $q_t = [w\ \mathbf{v}]$, which results in the derivatives becoming

$$\frac{\partial d_{\mathrm{rot}}}{\partial q_t} = 2\left[w\mathrm{d} + \mathbf{v} \times \mathrm{d} \;\middle|\; \mathbf{v}^T \mathrm{d} I_3 + \mathbf{v}\mathrm{d}^T - \mathrm{d}\mathbf{v}^T - w[\mathrm{d}]_\times \right]. \tag{109}$$

$$\Phi = \begin{bmatrix} I & -Rv_x I\Delta t - \frac{Rv_x[\omega]_\times}{\|\omega\|^2}\left(R\{\omega\Delta t\}^T - I + [\omega]_\times \Delta t\right) & \frac{1}{2}Rv_x R\{w\Delta t\}^T \Delta t^2 & -RU\Delta t \\ 0 & R\{\omega\Delta t\}^T & -I\Delta t - \frac{[\omega]_\times}{\|\omega\|^2}\left(R\{\omega\Delta t\}^T - I + [\omega]_\times \Delta t\right) & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix} \tag{82}$$

The complete derivation of the quaternion Jacobian can be referenced in [10, Sec. 4.3.2].

4) Least Squares Regression: The system used the standard least squares regression approach to estimate the triangulation method. The sensors that this research uses are UWB decawave sensors; these sensors return one-way time-of-flight range measurements and their current position. Once the range measurement was recorded, the position vector of the anchors is annotated as $\boldsymbol{p}_{a,i,k} = [x_{a,i,k}, y_{a,i,k}, z_{a,i,k}]$, while the target position of the robot is denoted as $\boldsymbol{p}_{r,k} = [x_{r,k}, y_{r,k}, z_{r,k}]$. The distance between two vectors is defined as $r_{i,k} = \|\boldsymbol{p}_{r,k} - \boldsymbol{p}_{a,i,k}\|_2$, and the true measurement as $\hat{r}_{i,k}$. The least squares problem could be formulated as follows:

$$\boldsymbol{p}_{r,k} = \min_{p_{r,k}} \sum_{i=1}^{N} (r_{i,k} - \hat{r}_{i,k})^2. \tag{110}$$

The closed-form solution of the least squares regression problem follows from the standard implementation of multilateration using least squares [16], [25], [26]:

$$\boldsymbol{p}_{r,k} = \frac{1}{2} A_k^{\dagger} b_k \tag{111}$$

$$A_k = \begin{bmatrix} x_{a,1,k} - x_{a,2,k} & y_{a,1,k} - y_{a,2,k} & z_{a,1,k} - z_{a,2,k} \\ \vdots & \vdots & \vdots \\ x_{a,1,k} - x_{a,N,k} & y_{a,1,k} - y_{a,N,k} & z_{a,1,k} - z_{a,N,k} \end{bmatrix} \tag{112}$$

$$b_k = \begin{bmatrix} \hat{r}_{2,k} - \hat{r}_{1,k} + \|\boldsymbol{p}_{a,1,k}\|_2^2 - \|\boldsymbol{p}_{a,2,k}\|_2^2 \\ \vdots \\ \hat{r}_{N,k} - \hat{r}_{1,k} + \|\boldsymbol{p}_{a,1,k}\|_2^2 - \|\boldsymbol{p}_{a,N,k}\|_2^2 \end{bmatrix}. \tag{113}$$

For the regression to produce a valid solution to the regression output, the rank$(A) \geq \dim(\boldsymbol{p}_r)$, where runtime complexity of the least squares regression is $\mathcal{O}(N)$. As such, the sensor input z_k was defined below, where p_k was the state position

$$z_k = \left[\delta p_k\right] = \left[p_{r,k} - p_k\right]. \tag{114}$$

B. $X_{\delta x}$ Derivation

The second part, $X_{\delta x} = (\partial x_t / \partial \delta x)|_x$, was the Jacobian of the true state concerning the error state. This part could be derived here as it depended on the SHFAF's states [10]

$$X_{\delta x} = \begin{bmatrix} \frac{\partial(p+\delta p)}{\partial \delta p} & & & 0 \\ & \frac{\partial(q \otimes \delta q)}{\partial \delta \theta} & & \\ & & \frac{\partial(\omega_b+\delta\omega_b)}{\partial \delta \omega_b} & \\ 0 & & & \frac{\partial(v_b+\delta v_b)}{\partial \delta v_b} \end{bmatrix}. \tag{115}$$

which resulted in all identity blocks, except for the 4×3 quaternion term $Q_{\delta\theta} = ((\partial(q \otimes \delta q))/(\partial \delta\theta))$, which created the following form:

$$X_{\delta x} = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & Q_{\delta\theta} & 0 \\ 0 & 0 & I_1 \end{bmatrix}. \tag{116}$$

Using the chain rule, and the fact that $q_1 \otimes q_2 = [q_1]_L q_2$, where

$$[q]_L = \begin{bmatrix} q_w & -q_x & -q_y & -q_z \\ q_x & q_w & -q_z & q_y \\ q_y & q_z & q_w & -q_x \\ q_z & -q_y & q_x & q_w \end{bmatrix} \tag{117}$$

and the limit $\delta q \to_{\delta\theta \to 0} \begin{bmatrix} 1 \\ (1/2)\delta\theta \end{bmatrix}$

$$Q_{\delta\theta} = \left.\frac{\partial(q \otimes \delta q)}{\partial \delta\theta}\right|_q = \left.\frac{\partial(q \otimes \delta q)}{\partial \delta q}\right|_q \left.\frac{\partial \delta q}{\partial \delta\theta}\right|_{\delta\theta \hat{=} 0} \tag{118}$$

$$= \left.\frac{\partial([q]_L \delta q)}{\partial \delta q}\right|_q \left.\frac{\partial \begin{bmatrix} 1 \\ \frac{1}{2}\delta\theta \end{bmatrix}}{\partial \delta\theta}\right|_{\delta\theta \hat{=} 0} \tag{119}$$

$$= [q]_L \frac{1}{2} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{120}$$

$$= \frac{1}{2} \begin{bmatrix} -q_x & -q_y & -q_z \\ q_w & -q_z & q_y \\ q_z & q_w & -q_x \\ -q_y & q_x & q_w \end{bmatrix}. \tag{121}$$

IV. Update Nominal State

After the SHFAF update, the nominal state was updated with the observed error state, as $x \leftarrow \delta x \oplus \hat{\delta x}$ [10]

$$p \leftarrow p + \hat{\delta p} \tag{122}$$

$$q \leftarrow q \otimes q\{\hat{\delta\theta}\} \tag{123}$$

$$\omega_b \leftarrow \omega_b + \hat{\delta\omega_b} \tag{124}$$

$$v_b \leftarrow v_b + \hat{\delta v_b}. \tag{125}$$

V. SHFAF Reset

After the error injection into the nominal state, the error's covariance and state mean $\hat{\delta x}$ were updated [10].

The reset function $g(\cdot)$, was written as

$$\delta x \leftarrow g(\delta x) = \delta x \ominus \hat{\delta x} \tag{126}$$

where $\ominus$ was the inverse of $\oplus$. The reset was, thus,

$$\hat{\delta x} \leftarrow 0 \tag{127}$$

$$P \leftarrow GPG^T \tag{128}$$

where G was the Jacobian matrix defined by

$$G = \left.\frac{\partial g}{\partial \delta x}\right|_{\hat{\delta x}}. \tag{129}$$

Similar to the $X_{\delta x}$ Jacobian, this Jacobian had identity matrices along its diagonal except with the orientation error $((\partial \delta\theta^+)/(\partial \delta\theta)) = I - [(1/2)\hat{\delta\theta}]_\times$ [10]

$$G = \begin{bmatrix} I_3 & 0 & 0 \\ 0 & I - \left[\frac{1}{2}\hat{\delta\theta}\right]_\times & 0 \\ 0 & 0 & I_4 \end{bmatrix}. \tag{130}$$

VI. SHFAF Implementation

A. Prediction

The prediction step updates the state and error state of the SHFAF based on the odometry control inputs. First, the algorithm starts by updating the nominal state as explained in Section II-B. Following this, the error state prediction is performed as explained in Section II-E.

B. Outlier Detection and Correction

When computing the ratio between the expected measurement covariance and the theoretical covariance, the measurement input was deemed an outlier if the ratio exceeds a certain threshold ξ. The value ξ is derived by trial and error to optimize the filter's performance. The outlier was then iteratively corrected until the element converged to an acceptable value [16]

$$D_k = HPH^T + R + H\delta x \delta x^T H^T \tag{131}$$

$$Z_k = \mathbb{E}\left[z_{k|k-1} z_{k|k-1}^T\right] \tag{132}$$

$$= \sum_{j=k-l+1}^{k} \sigma_j z_{k|k-1} z_{k|k-1}^T \tag{133}$$

$$G_k = \mathbb{E}\left[z_k z_k^T\right] \tag{134}$$

$$= \mathbb{E}[\epsilon\epsilon^T] + \mathbb{E}\left[z_{k|k-1} z_{k|k-1}^T\right] \tag{135}$$

$$= \hat{S}_k + Z_k \tag{136}$$

$$M_k = \text{diagonal}(G_k)/\text{diagonal}(D_k) \tag{137}$$

$$fr_{i,k} = \begin{cases} 1, & M_{i,k} \le \xi_i \\ \dfrac{1}{\sqrt{M_{i,k}}}, & M_{i,k} > \xi_i \end{cases} \tag{138}$$

$$z_k' = fr_k \times z_k. \tag{139}$$

C. Update

The measurement update integrated the input z_k and updated the current state of the robot; however, if one of the measurement input's elements was determined to be an outlier, the system corrected the measurement's outlier [16]

$$\epsilon_k = z_k - H\delta x \tag{140}$$

$$S = HPH^T + R \tag{141}$$

$$\epsilon\epsilon^T = HPH^T + R, \quad \text{if converged then equal} \tag{142}$$

$$\hat{R} = \epsilon\epsilon^T - HPH^T \tag{143}$$

$$R_k = \left(1 - s_k^\alpha d_k\right) R_{k-1} + s_k^\alpha d_k \hat{R} \tag{144}$$

$$d_k = \frac{\lambda - b}{\lambda - b^{k+1}} \tag{145}$$

$$\sigma_j = a^{k-j} \frac{1-a}{1-a^l} \tag{146}$$

$$\hat{S}_k = \mathbb{E}[\epsilon\epsilon^T] = \sum_{j=k-l+1}^{k} \sigma_j \epsilon_j \epsilon_j^T \tag{147}$$

$$r_k = \left| \frac{\text{Tr}(\hat{S}_k)}{\text{Tr}(S_k)} - 1 \right| \tag{148}$$

$$s_k = \begin{cases} 1, & k \le k_s \\ \text{fuzzy}(r_k), & k > k_s. \end{cases} \tag{149}$$

D. State Update

The system's Kalman gain and update states were computed with the correct measurement input

$$K = PH^T(HPH^T + R)^{-1} \tag{150}$$

$$\hat{\delta x} \leftarrow K(y - h(\hat{x_t})) \tag{151}$$

$$P \leftarrow (I - KH)P(I - KH)^T + KRK^T \tag{152}$$

$$\hat{x_t} = x \oplus \hat{\delta x}. \tag{153}$$

Finally, the error state and covariance matrix are reset through the implementation described in Section V.

VII. SHFAF Modifications

A. R Estimate

Some issue that could be noticed through the R_k update is that the matrix $\hat{R}$ needed to be positive definite; however, from the formulation of the update, it could be noted that the system did not necessarily ensure the positive definiteness of the system. As such, $\epsilon\epsilon^T - HPH^T$ was not necessarily constrained to be positive definite. This system reformulated the $\hat{R}$ update to be [27]

$$\hat{R} = \epsilon\epsilon^T + HPH^T. \tag{154}$$

Another way to ensure the positive definiteness was through defining $\hat{R}$ as

$$\hat{R} = (I - KH)\epsilon\epsilon^T(I - KH)^T + HPH^T \tag{155}$$

thus correcting the issue where the estimate $\hat{R}$ could become negative. The issue of R divergence was common in the previous implementation of the SHFAF. It required tuning of the parameters and the initial P, R to ensure that the system was less likely to diverge.

B. $s_k^\alpha d_k$ Factor Constraint

Another issue arose with the new definition of the fuzzy innovation factor for the measurement noise matrix update. The desired range for s_k^α is $s_k^\alpha d_k \in [0, 1]$. However, the range could not always be guaranteed because there was a lack of assumptions on the fuzzy system's output and the hyperparameters d, α, λ. The fuzzy system output could be bounded in the range of $s_k \in [s_{\min}, s_{\max}]$, and that $s_k \ge 0$, $\alpha \in [0, 1]$, $\lambda \ge 1$, $b \in [0, 1)$, $\lambda > \beta$. The goal became to find the minimum k_s from (149), where the conditions hold. If such a k_s was impossible, the system's hyperparameters would be modified such that the output range would remain within the domain.

The upper bound was always greater than 0, given that $\lambda \ge b$ and $b \le 1$. If b. The solution was determined when the system equaled 1 to find min k_s. This was solved because $s_k^\alpha d_k$ decreases monotonically with k; as such, the minimum value of the system was 1

$$s_{k,\max}^\alpha \frac{\lambda - b}{\lambda - b^{k+1}} = 1$$

$$k = \frac{\log\left(s_{k,\max}^\alpha (b - \lambda) + \lambda\right)}{\log(b)} - 1.$$

Thus, $k_s = \max\{((\log(s_{k,\max}^{\alpha}(b-\lambda)+\lambda+\epsilon))/(\log(b))) - 1, K\}$, where K was a user parameter that people could arbitrary setup. Where $\epsilon \approx 0$, usually 10^{-3}, this was just for numerical stability so that the system did not end up at zero. From this constraint, it also showed that b had to be strictly greater than 0, and $s_k^{\alpha}(b-\lambda)+\lambda > 0$. Additionally, $s^{\alpha} \geq 0$ no matter what, given that $\alpha \in [0, 1]$

$$s_{k,\max}^{\alpha}(b-\lambda)+\lambda > 0 \tag{156}$$
$$s_{k,\max}^{\alpha}b + \lambda(1-s^{\alpha}) > 0 \tag{157}$$
$$s_{k,\max}^{\alpha}b > -\lambda(1-s^{\alpha}) \tag{158}$$
$$b > \lambda\left(1 - s_{k,\max}^{-\alpha}\right) \tag{159}$$
$$> \lambda\left(1 - s_{\max}^{-\alpha}\right) \tag{160}$$
$$b = \min\left(1, \min\left(\lambda^{\frac{1}{k+1}}, \max\left(b, \lambda\left(1 - s_{\max}^{-\alpha}\right)\right)\right)\right). \tag{161}$$

The constraint $\lambda^{(1/k+1)}$ came from $\lambda - b^{k+1} \geq 0$, and the constraint $\lambda(1 - s_{\max}^{-\alpha})$ came from $s_{k,\max}^{\alpha}(b-\lambda)+\lambda \geq 0$.

If $b = 1$, set $\alpha = ((\ln(\lambda/\lambda - b))/(\ln(s_{\max})))$ and then recompute b using the new α.

C. M Update

It is noted that this article changed the recursive reweighting for the outlier features. The reweighting factor was modified to be $(1/M_{i,k})$ instead of the previous work's $(1/\sqrt{M_{i,k}})$ update [16]; the use of the linear $M_{i,k}$ instead of the $\sqrt{M_{i,k}}$ helps provide a faster convergence of the outlier features, as the outliers $M_{i,k} \geq 1$, $(1/M_{i,k}) \geq (1/\sqrt{M_{i,k}})$. Thus, given that the ratio between the convergence rates is $\sqrt{M_{i,k}}$, less iteration was required for the correction step. In addition, the linear outlier feature weighting causes more prominent outliers to have a more negligible impact on the filter, thus improving the performance.

1) Iteration Count Bounds: We could analyze the expected upper bound on the number of outlier iterations based on the different correction factors. We denoted the error in the measurement at the ith iteration by e_i. Writing the new correction, the iterative update can be modeled as a geometric decay. Convergence was declared when the M_k was below a tolerance ξ, i.e., $M_k \leq \xi$, for some iteration I. Starting with an initial error $e^{(0)}$, after I iterations we have

$$e^{(i+1)} \leq \frac{1}{M_k^{(i)}} e^{(i)} \tag{162}$$

and for outliers, $M_k^{(i)} > \xi > 1$. In the worst case

$$e^{(i+1)} \leq \frac{1}{\xi} e^{(i)}. \tag{163}$$

Thus, after I iterations

$$e^{(I)} \leq \left(\frac{1}{\xi}\right)^{I} e^{(0)}. \tag{164}$$

Convergence (i.e., $e^{(I)} \leq \epsilon$) required

$$\left(\frac{1}{\xi}\right)^{I} e^{(0)} \leq \epsilon \tag{165}$$

$$I \geq \frac{\log\left(e^{(0)}/\epsilon\right)}{\log(\xi)}. \tag{166}$$

For the alternative update using $1/\sqrt{M_k}$, we had

$$e^{(i+1)} \leq \frac{1}{\sqrt{\xi}} e^{(i)} \tag{167}$$

so that after I iterations

$$e^{(I)} \leq \left(\frac{1}{\sqrt{\xi}}\right)^{I} e^{(0)} \tag{168}$$

which implied

$$I \geq \frac{\log\left(e^{(0)}/\epsilon\right)}{\log(\sqrt{\xi})} = \frac{2\log\left(e^{(0)}/\epsilon\right)}{\log(\xi)}. \tag{169}$$

Thus, the upper bounds on the number of iterations were

$$I_{1/M} = \mathcal{O}\left(\frac{\log(e^{(0)}/\epsilon)}{\log(\xi)}\right) \tag{170}$$
$$I_{1/\sqrt{M}} = \mathcal{O}\left(\frac{2\log(e^{(0)}/\epsilon)}{\log(\xi)}\right). \tag{171}$$

This demonstrated that the $1/\sqrt{M}$ outlier correction term would take approximately twice the number of iterations as the $1/M$ outlier correction term.

VIII. Fuzzy Logic Model Optimization

The rules for the SHFAF system were defined as

If $r_k \in$ **Less**, then $s_k \in$ **Less**
If $r_k \in$ **Equal**, then $s_k \in$ **Equal**
If $r_k \in$ **More**, then $s_k \in$ **More**.

The fuzzy logic system was defined as a fuzzy inference system based on the ANFIS structure defined in [24]

$$W = A(T(\cdot)) \tag{172}$$
$$y = P^T Q \frac{W}{\|W\|_1}. \tag{173}$$

As such

$$P = \begin{bmatrix} O_{\text{Less}} & O_{\text{Equal}} & O_{\text{More}} \end{bmatrix}^{\text{T}} \tag{174}$$
$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = I_3 \tag{175}$$

where P was the output of triangular membership functions, given that the system uses a center-of-maximum defuzzification method for triangular membership functions, only the triangle vertices are necessary variables. The Q matrix represented the mapping between the input rule weights and the output membership functions. Given the system, a special derivation for A could be devised, simplifying the operation further

$$A(T) = \begin{bmatrix} \text{Less} & \text{Equal} & \text{More} \end{bmatrix}^{\text{T}}. \tag{176}$$

The system used triangular membership functions with a center-of-maximum defuzzification function and product T-norms. The function $A(T)$ was defined as the membership function outputs as triangles below.

A. A(T) Linearization

The $T(x)$ vector was rewritten as the vectorized sides of each of the triangles for the input membership functions, as illustrated in [24], where $t_{\text{down},i}$ represented the right side of the ith triangular membership function and $t_{\text{up},i}$ the left side. These portions were represented as the linear equations $y = mx + b$ to represent the sides of the triangles

$$T = \begin{bmatrix} t_{\text{down},1}(x) \\ t_{\text{up},2}(x) \\ t_{\text{down},2}(x) \\ t_{\text{up},3}(x) \end{bmatrix}. \tag{177}$$

The saturation function $\text{sat}(x) : \mathbb{R} \to [0, 1]$ was defined as a simple example $\text{sat}(x) = \min(\max(x, 0), 1)$. As such, $A(T)$ for this special case could be rewritten as

$$A(T) = \begin{bmatrix} \text{sat}(t_{\text{down},1}(x)) \\ \text{sat}(t_{\text{up},2}(x)) * \text{sat}(t_{\text{down},2}(x)) \\ \text{sat}(t_{\text{up},3}(x)) \end{bmatrix} \tag{178}$$

$$= \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{S_3} \text{sat}\left(\begin{bmatrix} t_{\text{down},1}(x) \\ t_{\text{up},2}(x) \\ t_{\text{down},2}(x) \\ t_{\text{up},3}(x) \end{bmatrix} \right) - \underbrace{\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}}_{Y_3}. \tag{179}$$

This can be written as such because $\text{sat}(t_{\text{up}}) * \text{sat}(t_{\text{down}}) = \text{sat}(t_{\text{up}}) + \text{sat}(t_{\text{down}}) - 1$.

For a general system, the system of matrices followed the symbolic pattern demonstrated below. For S_n and Y_n for $n \geq 2$, where n was the number of input and output triangles, $T \in \mathbb{R}^{2n-2}$, $and\, S_n \in \mathbb{R}^{n \times 2n-2}$

$$S_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{180}$$

$$S_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{181}$$

$$S_4 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{182}$$

$$Y_{n,i} = \begin{cases} 0, & i = 1 \text{ or } i = n \\ 1, & \text{else.} \end{cases} \tag{183}$$

The three parametric vectors were as follows:

$$l_{\text{right}} = \begin{bmatrix} l_{1,2} & l_{2,2} & l_{3,2} & \cdots & l_{n-1,2} \end{bmatrix} \in \mathbb{R}^{n-1} \tag{184}$$

$$l_{\text{left}} = \begin{bmatrix} l_{2,1} & l_{3,1} & l_{4,1} & \cdots & l_{n,1} \end{bmatrix} \in \mathbb{R}^{n-1} \tag{185}$$

$$\beta = \begin{bmatrix} \beta_1 & \beta_2 & \beta_3 & \cdots & \beta_{n-1} \end{bmatrix} \in \mathbb{R}^{n-1}. \tag{186}$$

The goal was to maintain the following constraints to create valid triangular membership functions. $l_{\text{right}}, l_{\text{left}}, \beta \geq 0$, as well as $\beta \leq l_{\text{right}}$ and $l_{\text{left}} \geq \beta$. The parameters l_{right} represented the left and right location vertices of the triangular membership function on the x-axis, while β represented the center vertex of the triangle. As such, the parameters were reparameterized. The exponential function was utilized to ensure the positive definiteness of the parameters

$$\hat{l}_{\text{right},i} = e^{l_{\text{right},i}} \tag{187}$$

$$\hat{\beta}_i = \sigma(\beta_i)\hat{l}_{\text{right},i} \tag{188}$$

$$\sigma(x) \in [0, 1] \tag{189}$$

$$= \frac{1}{1 + e^x} \tag{190}$$

$$\hat{l}_{\text{left},i} = \hat{\beta}_i + e^{l_{\text{left},i}}. \tag{191}$$

The vectors were stacked, and a row reordering of the vector was performed to ensure that the elements were in the proper format of alternating right and left components

$$l = \begin{bmatrix} \hat{l}_{\text{right}} \\ \hat{l}_{\text{left}} \end{bmatrix} \tag{192}$$

$$\hat{l} = T_l l \tag{193}$$

$$T_l = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{2n-2 \times 2n-2} \tag{194}$$

$$T_{\hat{l}} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & 0 & \cdots & 0 \\ & & \cdots & & & \\ 1 & 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{2n-2 \times 2n-2} \tag{195}$$

$$T_{\hat{\beta}} = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 1 & 0 & 0 & \cdots & 0 \\ & & \cdots & & & \\ 1 & 1 & 1 & 1 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{2n-2 \times n-1} \tag{196}$$

$$p = c + T_{\hat{l}}\hat{l} - T_{\hat{\beta}}\hat{\beta} \tag{197}$$

where $c \in \mathbb{R}$ was the leftmost side of the input membership functions, given that in the case of SHFAF, there were only three triangles, the full system could be explicitly defined as follows:

$$T(x) = \begin{bmatrix} t_{\text{down},0}(x) \\ t_{\text{up},1}(x) \\ t_{\text{down},1}(x) \\ t_{\text{up},2}(x) \end{bmatrix}$$

$$= \begin{bmatrix} \dfrac{(c + \hat{l}_{0,2}) - x}{\hat{l}_{0,2}} \\ \dfrac{x - (c + \hat{l}_{0,2} - \hat{\beta}_0)}{\hat{1}_{1,1}} \\ \dfrac{(c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2}) - x}{\hat{l}_{1,2}} \\ \dfrac{x - (c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2} - \hat{\beta}s_1)}{\hat{l}_{2,1}} \end{bmatrix} \tag{198}$$

$$= \begin{bmatrix} -x + (c + \hat{l}_{0,2}) \\ x - (c + \hat{l}_{0,2} - \hat{\beta}_0) \\ -x + (c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2}) \\ x - (c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2} - \hat{\beta}_1) \end{bmatrix} \oslash \begin{bmatrix} \hat{l}_{0,2} \\ \hat{l}_{1,1} \\ \hat{l}_{1,2} \\ \hat{l}_{2,1} \end{bmatrix}. \tag{199}$$

Simplified further, $E_n = [-1\ 1\ -1\ 1]^{\mathrm{T}}$. The simple rule for E_n is $E_{n,i} = (-1)^i\ \forall i \in \{1, \ldots, n\}$, which defines $\mathrm{diag}(E_n) = D_n$

$$T(x) = \left(E_n x + D_n \begin{bmatrix} c + \hat{l}_{0,2} \\ c + \hat{l}_{0,2} - \hat{\beta}_0 \\ c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2} \\ c + \hat{l}_{0,2} - \hat{\beta}_0 + \hat{l}_{1,1} + \hat{l}_{1,2} - \hat{\beta}_1 \end{bmatrix} \right) \oslash \begin{bmatrix} \hat{l}_{0,2} \\ \hat{l}_{1,1} \\ \hat{l}_{1,2} \\ \hat{l}_{2,1} \end{bmatrix} \tag{200}$$

$$T(x) = (E_n x - D_n p) \oslash \hat{l} \tag{201}$$

$$= \left(E_n x - D_n \left(c + T_{\hat{l}} \hat{l} - T_{\hat{\beta}} \hat{\beta} \right) \right) \oslash \hat{l} \tag{202}$$

$$= \left(E_n (x - c) - D_n T_{\hat{l}} \hat{l} + D_n T_{\hat{\beta}} \hat{\beta} \right) \oslash \hat{l}. \tag{203}$$

Remember that $x \in \mathbb{R}$ is a scalar and p is a vector.

The simplified $A(T)$ was thus now

$$A(T(x)) = S_n \mathrm{sat} \left((E_n x - D_n p) \oslash \hat{l} \right) - Y_n. \tag{204}$$

B. P Constraint

There was a need to ensure that the output of the ANFIS was always positive and monotonically increasing. To ensure this, constraints were imposed on the P parameters. As such P's variables O_{Less}, O_{Equal}, O_{More} were reparameterized

$$\hat{O}_{\mathrm{Less}} = O_{\mathrm{Less}}^2 \tag{205}$$

$$\hat{O}_{\mathrm{More}} = \hat{O}_{\mathrm{Less}} + O_{\mathrm{More}}^2 \tag{206}$$

$$\hat{O}_{\mathrm{Equal}} = \sigma(O_{\mathrm{Equal}})(\hat{O}_{\mathrm{More}} - \hat{O}_{\mathrm{Less}}) + \hat{O}_{\mathrm{Less}} \tag{207}$$

$$\hat{P} = \begin{bmatrix} \hat{O}_{\mathrm{Less}} & \hat{O}_{\mathrm{Equal}} & \hat{O}_{\mathrm{More}} \end{bmatrix}^{\mathrm{T}}. \tag{208}$$

C. Softplus Saturation

The saturation function $\mathrm{sat}(x) = \min(\max(x, 0), 1)$ could be replaced to approximate the piecewise linear characteristics with a continuous function. One way to approximate a ReLU was through a Softplus function $s(x) = \ln(1 + e^x)$. However, this function could be tuned to be sharper on the edges to act closer to a hard saturation function. The function definition with its derivative follows:

$$s(x) = a_1 \left(\ln \left(1 + e^{a_2 x} \right) - \ln \left(1 + e^{a_2 (x-1)} \right) \right) \tag{209}$$

$$\frac{\partial}{\partial x} s(x) = \frac{1}{2} \sinh \left(\frac{a_2}{2} \right) \mathrm{sech} \left(\frac{a_2 x}{2} \right) \mathrm{sech} \left(\frac{a_2}{2} (1 - x) \right) \tag{210}$$

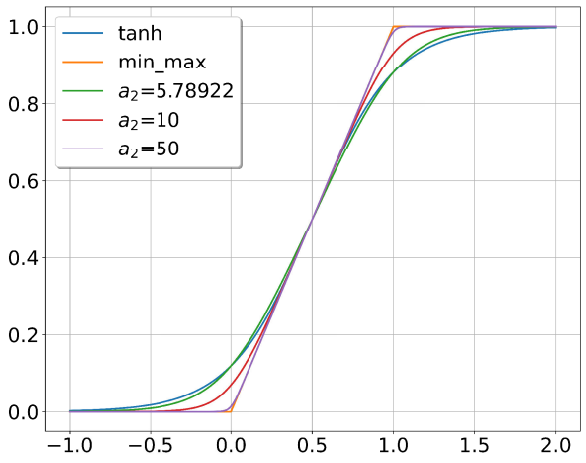


Fig. 1. Saturation function comparison.

$$\frac{\partial}{\partial a_2} s(x) = \frac{e^{a_2}(x-1)}{a_2(e^{a_2 x} + e^{a_2})} - \frac{x}{a_2(e^{a_2 x} + 1)} + \frac{1 - s(x)}{a_2}. \tag{211}$$

Fig. 1 depicted the different types of saturation functions. The system needed constraints such that it would always be centered such that $s(1/2) = (1/2)$ and acted similarly to the sigmoid and the current saturation function. This gave the constrain that assuming $a_1, a_2 > 0$

$$a_1 = \frac{1}{a_2} \tag{212}$$

where, as $a_2 \to \infty$ or $a_1 \to 0$, the system converged to a min–max formulation of the saturation function. For numerical stability with small values of a_1, training a_2 was easier. Further restricting what a_2 could be ensured that the saturation function did not diverge significantly from the intended saturation function. The term a_2 was constrained such that the saturation function followed a $\tanh(x)$ equivalent of the saturation function

$$\frac{\tanh(2x - 1) + 1}{2} = \frac{1}{a_2} \ln \left(\frac{1 + e^{a_2 x}}{1 + e^{a_2 (x-1)}} \right). \tag{213}$$

The smooth saturation function was evaluated at $x = 0$ or $x = 1$ to find a close approximation of the function

$$\frac{1}{1 + e^2} = \frac{\ln(2) - \ln(1 + e^{-a_2})}{a_2}. \tag{214}$$

Which returned an approximate a_2 minimum of 5.78922. To ensure that the gradients are not too large or too small, the function's maximum was constrained to be a maximum of 50. As such, $a_2 \in [5.78922, 50]$ was constrained. When training the ANFIS, the algorithm imposed a sigmoidal constraint for a_2, i.e., $\hat{a}_2 = \sigma(a_2)(a_{\max} - a_{\min}) + a_{\min}$.

IX. Experimentation

A. Comparison

This article compared three SHFAF models. VelSHFAF represented this article's implementation of the SHFAF algorithm;

Fig. 2. Clearpath Jackal robot configuration.

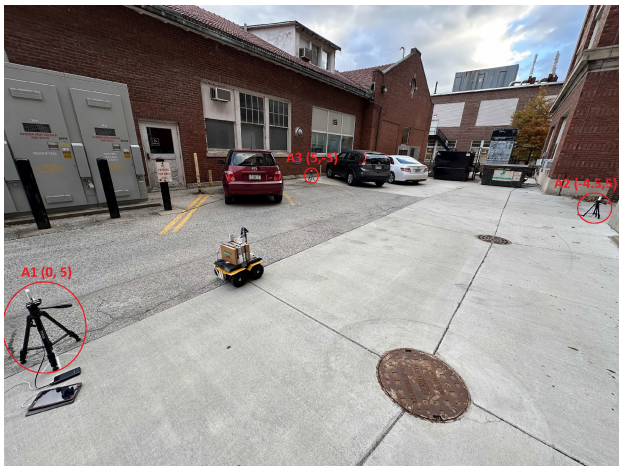


Fig. 3. Environment setup.

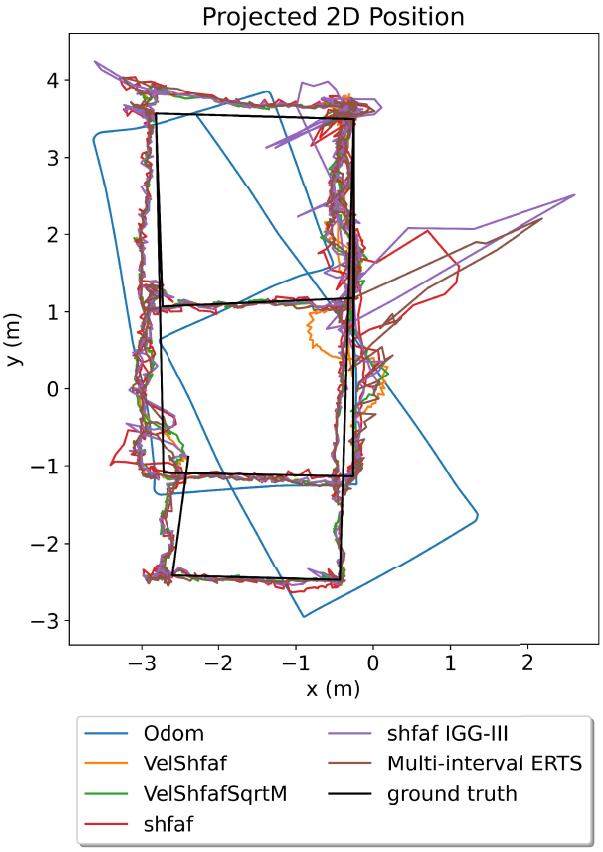


Fig. 4. Projected 2-D position estimation.

VelSHFAFSqrtM uses the same algorithm; however, it uses the square root update formulation for the outlier reweighting and, finally, the original SHFAF from [16]. The Clearpath Jackal robot was used with an onboard VectorNav IMU, wheel encoders, and two Decawave UWB sensors (Fig. 2). The two UWB sensors on the robot were set up with one being configured as an anchor and the other as a tag. The role of the tag on the robot was to enable the trilateration of the robot from the other anchors in the terrain, while given its offset position, being able to indirectly provide additional information about the robot's heading compared to placing it in the center of rotation of the system, as given that these sensors are omnidirectional, in-place rotations cannot be picked up from just trilateration. The additional anchor on the robot allowed the system, once it has been localized to a sufficient degree (time-based or covariance-based), to start broadcasting its estimated position to the other nodes in the network, thus enlarging the localization area. However, it should be noted that this can cause a positive feedback loop to appear due to compounding errors being propagated. The UWB sensors were tripod-mounted Decawave DWM1001-DEV modules using DWM100 ICs broadcasting through channel 5, which claims a $<$15 cm accuracy and could maintain a connection for up to a range of 60 m. The Clearpath Jackal followed the paths shown in the coming diagrams and was controlled via teleoperation.

There were three sensors set in the environment located at the positions (0, 5, 0 m), ($-$4.5, 5, 0 m), and (5, 5, 0 m), annotated as A1, A2, and A3, respectively, as depicted in Fig. 3.

The graphs demonstrate the robot's localization using the three algorithms plotted.

From the 2-D projected position estimated graph (Fig. 4), it could be noted that the raw odometry data from onboard encoders and IMU had a significant drift over time. However, all the SHFAF implementations managed to follow the correct trajectory approximately, with no significant insight was gained from the graphs as they all follow similar trajectories. However, it could be noted that once the systems were viewed in 3-D in Fig. 5, the SHFAF had issues where the system suddenly spikes in the z-coordinate, jumping to near the $-$3.5 m mark, and taking a significant amount of time before correcting itself in the z-coordinate of 0 m. In addition, when the SHFAF was in the $-$3.5 m range, it could be noted that a significant position jump was observed due to one of the outlier measurements, causing the system to jump further in its position estimation. In contrast, the VelSHFAF and the

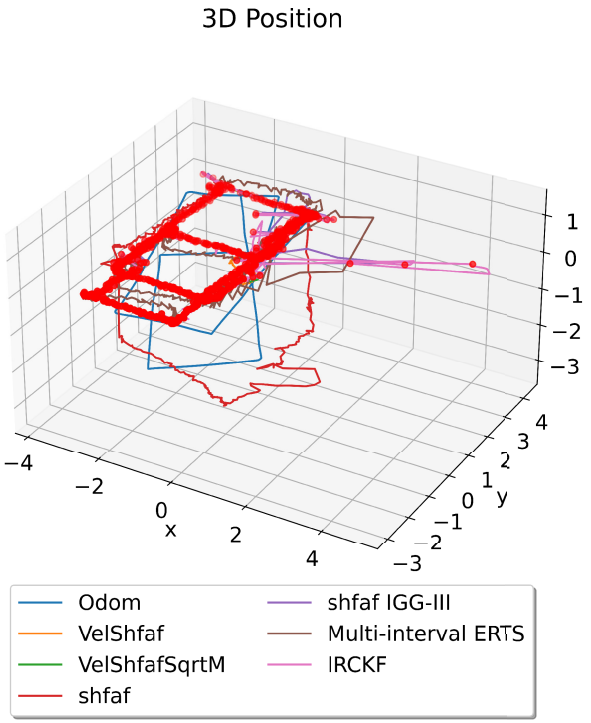


Fig. 5. 3-D position estimation.

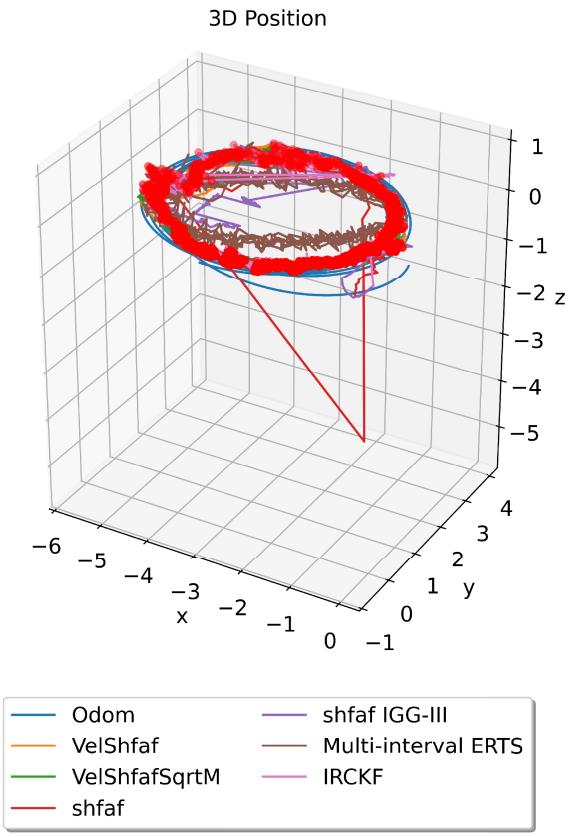


Fig. 6. Circular 3-D position estimation.

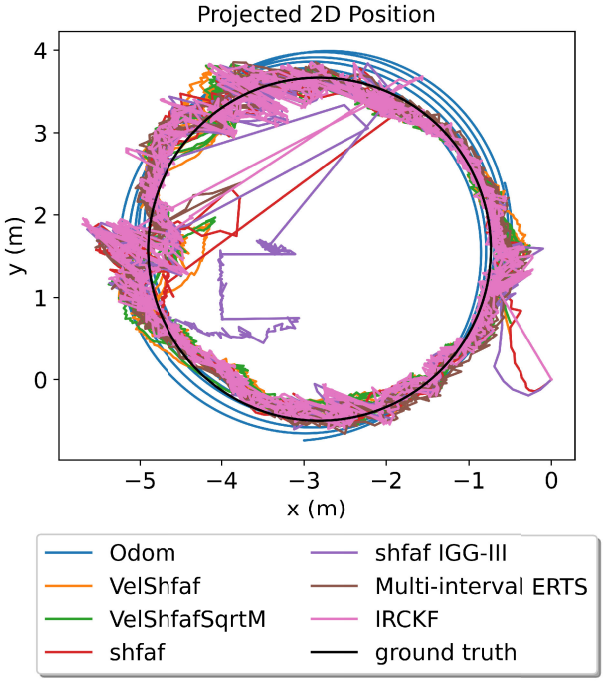


Fig. 7. Circular projected 2-D position estimation.

VelSHFAFSqrtM did not have this observed discrepancy and remained in the correct plane. The VelSHFAF observed a jump at the (−0.5, 0.5 m) mark, most likely due to an outlier; however, it quickly corrected the estimation. VelSHFAFSqrtM, in that regard, was more robust to the outlier. The sudden jumps in the z-axis by the SHFAF were noticed in both trajectories in Figs. 6 and 7.

From Table I, VelShfaf had the smallest position RMSE between the ground truth and the estimated position. Given that no ground-truth sensors were used on the robots (such as RTK), the ground truth used were derived manually, for the circle trajectory given that the robot was tasked to move in a constant linear and angular velocity of 1 and 0.5 rad/s, respectively, and the expected ground truth was generated from its resultant circle. For the grid task, the paved terrain that was tested had measured dimensions across which the robot was manually driven. The followed path was thus measured and drawn as ground truth. Minor deviations from the ground truth due to deviations from manual driving or terrain would realistically not affect the final results, as all the algorithms would utilize the same data. The UWB trilateration was also used to cross-correlate the ground-truth positions. This newer VelSHFAF thus demonstrates a performance increase of ≈33% in the worst case compared to the original SHFAF algorithm. Compared to more modern UWB localization algorithms such as the multiinterval extended finite impulse response-based Rauch–Tung–Striebel (multiinterval ERTS) [23] and the improved robust cubature Kalman filter (IRCKF) [22] or just the SHFAF with the addition of IGG-III [28], the VelSHFAF and VelSHFAFSqrtM perform ≈40% better on the grid test course and ≈1% better on the circle course. In the circle course, the results are similar between the different algorithms, and the VelSHFAF performs worse than the other modern techniques.

An issue of increased computational cost arises with the enhanced accuracy of the closed-form solution calculation

TABLE I
COMPARISON RESULTS

RMSE Metric	Grid	Circle
Uwb	0.2204	0.1699
Odom	0.7331	0.215
Shfaf	1.2894	0.230
VelShfaf	**0.1396**	0.1808
VelShfafSqrtM	0.1347	**0.1728**
Shfaf + IGG-III	0.1955	0.2236
Multi-interval ERTS	0.2824	0.1753
IRCKF	0.280	0.1763

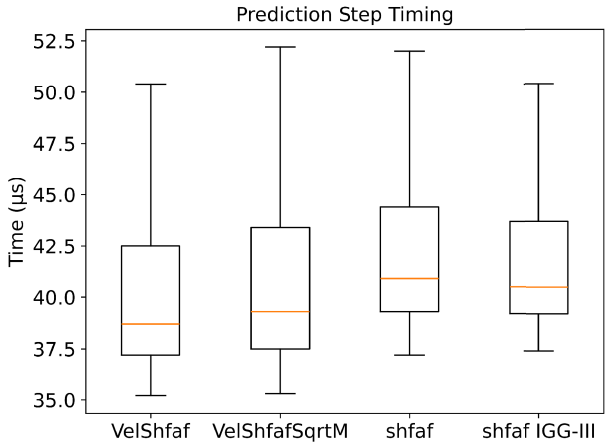


Fig. 8. Predict time.

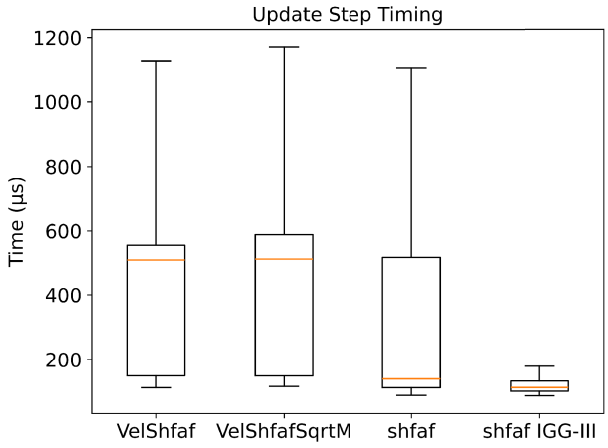


Fig. 9. Update time.

and state reset calculation. This article ran these intensive calculations on a machine with an i9-13900k CPU, and the computation times for the respective calculations were listed in Figs. 8 and 9.

As noted through the box plots, the algorithms had similar running times for the prediction steps, with all running within 2 ns; however, the measurement update step follows a different story. It could be noticed that the SHFAF had a faster computation speed with a median of 108 μs compared to the respective 213 and 216 μs update steps of the VelSHFAF and the VelSHFAFSqrtM, making it run twice as fast; however, given that the system was running in the order of nanoseconds, the decrease in computation speed was deemed

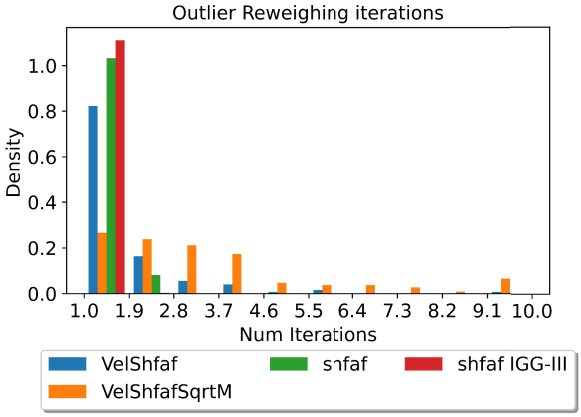


Fig. 10. Update time.

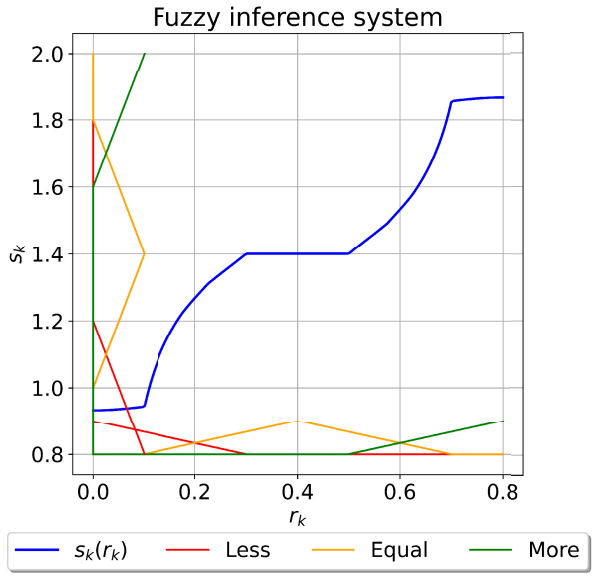


Fig. 11. Fuzzy inference system.

appropriate given the increased robustness and the improved accuracy.

The primary attribution to the SHFAF's performance boost was the outlier reweighting step, shown in Fig. 10. The number of iterations performed by the SHFAF was distributed between 1 and 2. At the same time, the VelSHFAF had an exponentially decreasing reweighting step number, while the VelSHFAFSqrtM had a more uniform distribution. The outlier reweighting step contributes most of the computation time in the measurement update due to the recomputation of the weighted averages and the other variables. Thus, the fewer the number of iterations, the faster the steps. The measurement update time of the VelSHFAF was slightly less skewed to be lower than the VelSHFAFSqrtM for that reason, as illustrated in Fig. 9.

For the fuzzy logic system, the mapping from r_k to s_k was used, which is demonstrated in Fig. 11. The respective triangle membership functions of r_k are demonstrated at the bottom of

Algorithm 1 ANFIS Training

initialize θ_0
for epoch **in** num_epochs
  reinitialize VelSHFAF's $x, \delta x, P, R, Q$
  $l \leftarrow 0$
  for t **in** max_steps
    $\hat{x}_{t+1} \leftarrow \text{SHFAF}_{\theta_i}(x_t, u_t, z_t)$
    $l \leftarrow l + \frac{1}{2}\left\|x_{t+1} - \hat{x}_{t+1}\right\|_2^2$
  $\theta_{i+1} \leftarrow \theta_i - \mu \nabla_{\theta_i} l$

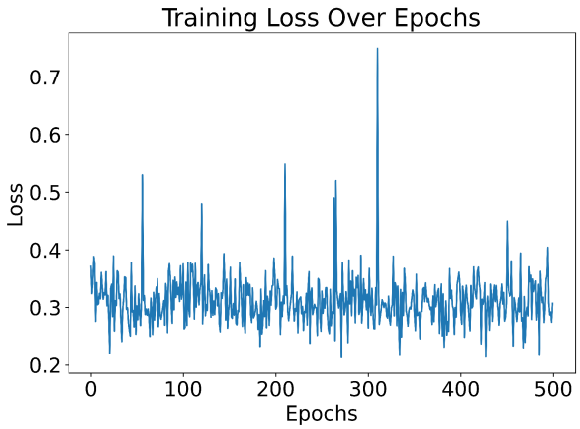


Fig. 12. ANFIS loss function over time.

the figure, and the output triangle membership functions of s_k are illustrated on the left side.

B. Offline ANFIS Training

Given the structure of the fuzzy inference system as a differentiable module, an attempt was made to train the ANFIS network's parameters θ defined as $\{c, l_{\text{left}}, l_{\text{left}}, \beta, a_2\}$ to improve the performance of the VelSHFAF model further. To perform this, the system posed the following optimization:

$$\hat{\theta} = \arg\min_{\theta} \frac{1}{2}\sum_{t=0}^{T-1} \|x_{t+1} - \text{VelSHFAF}(x_t, u_t, z_t)\|_2^2 \quad (215)$$

where $\text{VelSHFAF}(x_t, u_t, z_t)$ was the output state of the VelSHFAF Kalman filter based on the current state x_t, the control input u_t, and the measurement z_t, with the predicted update steps. Thus, the optimization was to minimize the loss over a sample trajectory. However, due to the recursive and nonlinear nature of the problem, there is no closed-form solution. If the system was a linear derivation, the closed form could have be derived as

$$(\hat{A}, \hat{B}) = \arg\min_{A,B} \frac{1}{2}\sum_{t=0}^{T-1} \|x_{t+1} - Ax_t - Bu_t\|_2^2 \quad (216)$$

$$(\hat{A}, \hat{B}) = \left(\sum_{t=0}^{T-1} x_{t+1} z_t^T\right)\left(\sum_{t=0}^{T-1} z_t z_t^T\right)^{-1}, \quad z = \begin{bmatrix} x_t \\ u_i \end{bmatrix}. \quad (217)$$

The attempt thus used gradient descent to tune the system's parameters demonstrated in Algorithm 1.

However, the training implementation did not demonstrate any improvements to the loss, as depicted in Fig. 12.

X. Conclusion

This article applied methods to improve the previous SHFAF implementation [16] by Liu et al., by using a more robust P update definition in the Joseph form and conditioning the noise covariance matrix update step with averaging coefficients for the system to remain numerically stable. The VelSHFAF was also redefined to utilize the fused wheel and IMU odometry information instead of relying on the integrated IMU measurements; the fused inputs provided a more accurate representation of the system's dynamics while improving issues with IMU noise.

In addition, a different definition of the predicted R matrix was used to keep the system from diverging into a negative definite system. This enabled a more robust VelSHFAF parameterization.

This article also introduced an optimized and differentiable implementation of the fuzzy logic system to train the system's R-averaging step based on the ANFIS architectures. A set of novel trainable saturation functions was generated for the ANFIS structure to facilitate smooth output responses and gradients for triangle-based membership functions. Minor performance improvements were observed during testing, although further training and evaluation of the ANFIS system are needed to fully exploit its capabilities.

Experimental validation was done using real-world data from a Clearpath Jackal robot equipped with Qorvo UWB sensors and static nodes. Regarding localization accuracy, the proposed VelSHFAF system outperformed the previous SHFAF implementation by approximately 30%–25% across two test courses, demonstrating its improved performance.

In future work, online training of the ANFIS system will be explored instead of offline optimization. This adaptive Kalman filter will also be combined with an adaptive mesh network of sensor nodes to perform multirobot localization. In addition, smoothing functions for the outputs will be explored to reduce the SHFAF's high-frequency output signals.

References

[1] A. Alarifi et al., "Ultra wideband indoor positioning technologies: Analysis and recent advances," *Sensors*, vol. 16, no. 5, p. 707, May 2016. [Online]. Available: http://www.mdpi.com/1424-8220/16/5/707

[2] B. Van Herbruggen et al., "Wi-PoS: A low-cost, open source ultra-wideband (UWB) hardware platform with long range sub-GHz backbone," *Sensors*, vol. 19, no. 7, p. 1548, Mar. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/7/1548

[3] R. S. Thoma, O. Hirsch, J. Sachs, and R. Zetik, "UWB sensor networks for position location and imaging of objects and environments," in *Proc. 2nd Eur. Conf. Antennas Propag. (EuCAP)*, 2007, p. 213. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/ic.2007.1336

[4] M. Tanenhaus, D. Carhoun, T. Geis, E. Wan, and A. Holland, "Miniature IMU/INS with optimally fused low drift MEMS gyro and accelerometers for applications in GPS-denied environments," in *Proc. IEEE/ION Position, Location Navigat. Symp.*, Apr. 2012, pp. 259–264.

[5] M. Ouyang, Z. Cao, P. Guan, Z. Li, C. Zhou, and J. Yu, "Visual-gyroscope-wheel odometry with ground plane constraint for indoor robots in dynamic environment," *IEEE Sensors Lett.*, vol. 5, no. 3, pp. 1–4, Mar. 2021.

[6] A. A. Housein, G. Xingyu, W. Li, and Y. Huang, "Extended Kalman filter sensor fusion in practice for mobile robot localization," *Int. J. Adv. Comput. Sci. Appl.*, vol. 13, no. 2, pp. 33–38, 2022. [Online]. Available: http://thesai.org/Publications/ViewPaper?Volume=13&Issue=2&Code=IJACSA&SerialNo=4

[7] W. Li and H. Leung, "Constrained unscented Kalman filter based fusion of GPS/INS/digital map for vehicle localization," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, vol. 2, Nov. 2003, pp. 1362–1367.

[8] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Trans. Autom. Control*, vol. 54, no. 6, pp. 1254–1269, Jun. 2009. [Online]. Available: http://ieeexplore.ieee.org/document/4982682/

[9] I. Brigadnov, A. Lutonin, and K. Bogdanova, "Error state extended Kalman filter localization for underground mining environments," *Symmetry*, vol. 15, no. 2, p. 344, Jan. 2023. [Online]. Available: https://www.mdpi.com/2073-8994/15/2/344/htm

[10] J. Solà, "Quaternion kinematics for the error-state Kalman filter," 2017, *arXiv:1711.02508*.

[11] Y. Zhou, C. Zhang, Y. Zhang, and J. Zhang, "A new adaptive square-root unscented Kalman filter for nonlinear systems with additive noise," *Int. J. Aerosp. Eng.*, vol. 2015, pp. 1–9, Oct. 2015. [Online]. Available: https://cir.nii.ac.jp/crid/1361699994820058496

[12] Y. Dong and Y. Wei, "Research on initial alignment for large azimuth misalignment angle with sage-husa adaptive filtering," *Infr. Laser Eng.*, vol. 42, pp. 2197–2201, 2013. [Online]. Available: http://www.irla.cn/en/article/id/340

[13] S. Wan-Xin, "Application of sage-husa adaptive filtering algorithm for high precision SINS initial alignment," in *Proc. 11th Int. Comput. Conf. Wavelet Actiev Media Technol. Inf. Processing(ICCWAMTIP)*, Dec. 2014, pp. 359–364.

[14] S. D. Brown and S. C. Rutan, "Adaptive Kalman filtering," *J. Res. Nat. Bur. Standards*, vol. 90, no. 6, p. 403, 1985. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/jres/090/jresv90n6p403_A1b.pdf

[15] H. Fang, M. A. Haile, and Y. Wang, "Robustifying the Kalman filter against measurement outliers: An innovation saturation mechanism," in *Proc. IEEE Conf. Decision Control (CDC)*, Sep. 2018, pp. 6390–6395. [Online]. Available: https://ieeexplore.ieee.org/document/8619140/

[16] J. Liu, J. Pu, L. Sun, and Z. He, "An approach to robust INS/UWB integrated positioning for autonomous indoor mobile robots," *Sensors*, vol. 19, no. 4, p. 950, 2019. [Online]. Available: http://www.mdpi.com/1424-8220/19/4/950

[17] H. E. Nyqvist, M. A. Skoglund, G. Hendeby, and F. Gustafsson, "Pose estimation using monocular vision and inertial sensors aided with ultra wide band," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Oct. 2015, pp. 1–10. [Online]. Available: http://ieeexplore.ieee.org/document/7346940/

[18] F. Matía, V. Jiménez, B. P. Alvarado, and R. Haber, "The fuzzy Kalman filter: Improving its implementation by reformulating uncertainty representation," *Fuzzy Sets Syst.*, vol. 402, pp. 78–104, Jan. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0165011419304865

[19] Z. Li, G. Chang, J. Gao, J. Wang, and A. Hernandez, "GPS/UWB/MEMS-IMU tightly coupled navigation with improved robust Kalman filter," *Adv. Space Res.*, vol. 58, no. 11, pp. 2424–2434, Dec. 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0273117716303982

[20] H. Wang, Y. Liu, W. Zhang, and J. Zuo, "Outlier-robust Kalman filter in the presence of correlated measurements," 2020, *arXiv:2011.13353*.

[21] L. Wang and S. Li, "Enhanced multi-sensor data fusion methodology based on multiple model estimation for integrated navigation system," *Int. J. Control, Autom. Syst.*, vol. 16, no. 1, pp. 295–305, Feb. 2018. [Online]. Available: https://link.springer.com/article/10.1007/s12555-016-0200-x

[22] Y. Li, Z. Gao, C. Yang, and Q. Xu, "A novel UWB/INS tight integration model based on ranging offset calibration and robust cubature Kalman filter," *Measurement*, vol. 237, Sep. 2024, Art. no. 115186.

[23] Y. Gao, W. Ma, J. Cao, J. Qu, and Y. Xu, "Range-only UWB SLAM for indoor robot localization employing multi-interval EFIR rauch-tung-striebel smoother," *Comput. Model. Eng. Sci.*, vol. 130, no. 2, pp. 1221–1237, 2022.

[24] M. Juston, S. Dekhterman, W. R. Norris, D. Nottage, and A. Soylemezoglu, "Hierarchical rule-base reduction based ANFIS with online optimization through DDPG," *IEEE Trans. Fuzzy Syst.*, vol. 32, no. 11, pp. 6350–6362, Aug. 2024.

[25] J. Liu, J. Pu, L. Sun, and Z. He, "An approach to robust INS/UWB integrated positioning for autonomous indoor mobile robots," *Sensors*, vol. 19, p. 950, 2019. [Online]. Available: https://doi.org/10.3390/s19040950

[26] S. Nawaz and N. Trigoni, "Robust localization in cluttered environments with NLOS propagation," in *Proc. 7th IEEE Int. Conf. Mobile Ad-hoc Sensor Syst. (IEEE MASS)*, Nov. 2010, pp. 166–175. [Online]. Available: http://ieeexplore.ieee.org/document/5663983/

[27] S. Akhlaghi, N. Zhou, and Z. Huang, "Adaptive adjustment of noise covariance in Kalman filter for dynamic state estimation," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Jul. 2017, pp. 1–5.

[28] Y. Yang, W. Gao, and X. Zhang, "Robust Kalman filtering with constraints: A case study for integrated navigation," *J. Geodesy*, vol. 84, no. 6, pp. 373–381, Jun. 2010. [Online]. Available: http://link.springer.com/10.1007/s00190-010-0374-6