Ad-Hoc Mesh Network Localization using Ultra-Wideband for Mobile Robotics

Marius F. R. Juston * and William R. Norris

Posted Date: 9 January 2024

doi: 10.20944/preprints202401.0684.v1

Keywords: ultra-wideband; unscented Kalman filter; Ad-Hoc localization

Article

Ad-Hoc Mesh Network Localization Using Ultra-Wideband for Mobile Robotics

Marius F. R. Juston *,† and William R. Norris †

University of Illinois Urbana-Champaign; wrnorris@illinois.edu
* Correspondence: mjuston2@illinois.edu; Tel.:+1-404-583-9452
† These authors contributed equally to this work.

Abstract: This article explores the implementation of high accuracy GPS denied Ad-Hoc localization. There is little research on Ad-Hock UWB-enabled localization systems with mobile and stationary nodes. This work aims to demonstrate the localization of bicycle-modeled robots in a non-static environment through a mesh network of mobile, stationary robots and ultra-wideband sensors. The non-static environment adds a layer of complexity when actors can enter and exit the node's field of view. The method starts with an initial localization step where each unmanned ground vehicle (UGV) uses the surrounding, available anchors to derive an initial local or, if possible, global position estimate. The initial localization uses a simplified implementation of the iterative multi-iteration Ad-Hoc Localization System (AHLos). This estimate was refined using an unscented Kalman filter (UKF) following a constant turn rate and velocity magnitude model (CTRV). The UKF fuses the robot's odometry and the range measurements from the Decawave ultra-wideband receivers stationed on the network nodes. Through this position estimation stage, the robot broadcasts to its neighbors its estimated position to help the others further improve their localization estimates and localize themselves. This wave-like cycle of nodes helping to localize each other allows the network to act as a mobile Ad-Hoc localization network.

Keywords: ultra-wideband (UWB); unscented Kalman filter (UKF); Ad-Hoc localization

1. Introduction

1.1. Overview

Due to the GPS' constant need for high accuracy and fidelity in global localization, issues arise when GPS access is limited in some environments. This could happen due to inadequate GPS accuracy or unavailability in indoor or urban environments such as tunnels [1,2]. Some sensors complement GPS-denied systems such as LIDARs, camera systems (mono or stereo), inertial measurement sensors, velocity sensors, altimeters, compasses, and more; however, most of the solutions are expensive or prone to more significant errors over time with no way to calibrate [3,4]. Ultra-wideband sensors are ideal for implementing a local GPS because they provide centimeter range accuracy, high range reliability, and low latency [5–7]. Despite GPS being affordable and reliable, several issues can make its use suboptimal. Issues include GPS being unavailable in indoor environments and only having five meters of precision, which is not ideal for a higher level of accuracy. This can be most readily noted in a lawn mower example, where 5 meters can be a substantial error relative to the robot's workspace area. Additional issues in a lawn-mower-type application are the potential of trees obstructing the GPS signal and the GPS's low update rate of 1 Hz, which is not sufficient to track fast vehicles. Thus, systems such as UWB, which can provide centimeter accuracy at higher rates, are more effective solutions. An RTK solution could be implemented to use centimeter-level accuracy for GPS; however, these are not cost-effective for such situations.

 doi:10.20944/preprints202401.0684.v1

1.1.1. Ultra-Wideband Sensors

The use of Ultra-wideband (UWB) in localization is a popular technology due to its accurate positioning capabilities, immunity against multipath fading, and resilience against active and passive interference's [5,8]. UWB sensors utilize a large frequency spectrum, from 3.1 to 10.6 GHz, and bandwidths of 500+ MHz to implement localization techniques. This means that UWB radar sensors can, compared to traditional optical or less powerful radar sensors, maintain their accuracy in more challenging terrain, such as indoor situations where multipath and interference errors are more prevalent. Thus, they retain the ability to sense and communicate in obstacle-heavy terrain and allow for more robust localization systems [9]. There are three node types in UWB sensors, the transmitters, which only transmit their operations; the anchor nodes, which are usually static with a set and verified location to help the other nodes in localizing; and mobile nodes, which contain a combination of transmitting antennas and receiving antennas.

1.2. Related Works

UWB localization problems were approached through different configurations of anchors and tags, combining stationary and mobile tags. Two main groups of UWB localization approaches were found in the current literature. The first approach, followed by [10–17], used a combination of initial location estimates using the UWB ranging measurements and a regression method followed by the use of a type of Kalman filter to further improve the estimate over time. The second approach followed a Mote Carlo-based simulation similar to particle filters [18].

To simplify communication issues and improve network robustness, decentralized sensors were most commonly utilized to perform the computation [16]. The decentralized initial location estimates follow a linear regression problem to solve triangulation or trilateration to get a robust position estimate [12,17]. However, the issue was that these systems assumed that the robot was constantly able to achieve communication with a minimum number of nodes before adequately operating. The robot was also assumed to be stationary during this period.

Current literature has explored more noise-robust extensions of Non-Linear Least Squares for triangulation. Systems using robust statistical techniques can mitigate the impact of outliers in measurements [19–21]. Some employ reformations of the reweighed least squares (IRLS) methods [22], while others transform the LSQ problem using a Majorization-Minimization (MM) approach [23]. While these methods usually look at static environments, some mobile beacon-based location methods track the anchor's messages and localizes themselves based on the message history [24,25]; however, even in these cases, the tracked nodes are static.

Multiple tags and anchors were placed on the robot to improve the understanding of its position and orientation estimates. This helped constrain its position and provided more reliability in its measurements [15,17]. An issue with previous work was that the systems localized each other in a relative sense; the nodes were localized relative to a base node. This was, however, not ideal for long-distance or multi-mesh scenarios as it did not ensure that each node was constrained to the same coordinate frame.

Once the position estimate was calculated, the system transitioned to position refinement techniques to improve further the time-varying sensor noise provided to the robot. The extended Kalman filter (EKF) and the unscented Kalman filter (UKF) have been used extensively in literature [15,16]. The UKF is sensitive to the inherent time invariant multipath effect and non-line-of-sight (NLOS) noise. To reduce the inherent error, different types of filter approaches have been used [6,10,14,26]. The noise increase was relatively significant for indoor scenarios because so many obstacles could block the direct line of sight of the UWB sensors. As such, using UWB sensors only for localization was not ideal. The noise inherent in the system was reduced, as demonstrated in this research, with the fusion of the wheel encoders and inertial measurement units (IMUs) to smooth out the position estimate.

 doi:10.20944/preprints202401.0684.v1

In contrast to the aforementioned methods, the second method for position estimation found in literature followed a Monte Carlo approach [18]. The algorithms followed a repeated random sampling process to arrive at a computationally convergent solution or solutions [27,28]. This sampling technique does provide possible issues with multiple equilibrium points and can be more computationally expensive than other options, depending on the number of sampling points. Efforts to enhance wireless network localization techniques have delved into integrating machine learning within the localization engine [29]. Semi-supervised particle swarms have been augmented to improve the data point selection; however, currently, these systems still assume that the anchor nodes are static [28]. The following section introduces this research problem formulation.

1.3. Contribution

This article focused on the localization of moving Unmanned Ground Vehicles (UGV) using stationary and moving UGV and Unmanned Aerial Vehicles (UAV). These vehicles had UWB sensors attached, enabling peer-to-peer ranging communication and allowing the robots to improve their respective localization using a cooperative network. The network helped provide a broad reach and create a local GPS network for the nearby robots.

The main contributions from the research are highlighted below:

1. Developed an initial localization framework where all agents could be non-stationary, and the UWB tags are offset from the center of the robot.
2. Designed a global/relative localization system based on the UKF for ground-based robots that could leverage ground and aerial range measurement data.
3. Created a pipeline for a mobile Ad-Hoc localization system.

The rest of the article was organized as follows. The environment and problem are described in the "Method" section, where the agents of the environment, the assumptions of the environment, and the detailed derivations of the proposed system are laid out. The "Simulation results" section contains the simulation results and interpretation. Finally, the work summary is presented in the "Conclusion and future work" section.

2. Method

The Ad-Hoc mesh network definition and implementation with the explanation of the information propagation throughout the network are explained. The following sections contributed to the initial position transform through a non-linear least squares regression, starting in Section 2.3 and ending in Section 2.4, followed by the position refinement using an unscented Kalman filter, starting in Section 2.5 and ending in Section 2.5.4.

2.1. Ad-Hoc Network

The proposed Ad-Hoc Mesh network followed a mobile decentralized, absolute localization Ad-Hoc system. The nodes in the network would consist of both stationary and mobile nodes. Due to the network's mobile nature, a decentralized network was more robust to the constantly changing topography of the network graph. The network was fine-grained (range-based) thanks to the incorporated sensors in this research, the UWB sensors, measuring the range using ToF (Time of Flight). The system was also required to converge to an absolute localization system; this means that the robot should be able to transition from an initial relative localization system, and once it had reached the required threshold for transitioning, convert the relative coordinate system to become a global localization problem.

The algorithm follows a similar but simplified version of the Iterative Multilateration AHLoS (Ad-Hoc Localization System) system [30]. The AHLoS system is a method to implement the localization of a mesh network in a flood-like fashion. The algorithm started with a graph that combined localized and unlocalized nodes; when an unknown node lies in the neighborhood of three

 doi:10.20944/preprints202401.0684.v1

or more anchors, the neighboring anchors' positions and distances were used to estimate its position. Once the position of an unknown node was estimated, the node became an anchor and thus continued the cycle until all the nodes were localized.

This article modified the AHLoS system to account for mobile nodes coming out and into the range of other localized nodes. The current and historical range measurements were only collected from the localized anchors and their associated position at every time step. The range measurements were then paired with the node's odometry measurement at the corresponding time. Linear extrapolation was employed if the odometry measurements were too far apart.

The localized nodes communicate to the target robot the range measurement and the localized node's current global position. Given the distributed nature of the mobile Ad-Hoc network, each robot would maintain a history of measurement data from each robot. With nodes frequently entering and exiting the robot's view and with histories stored locally, the system retains all information and remains unaffected by network topography changes. Given that the range measurement uses ToF for its UWB range estimation, no synchronization between nodes is required, thus making communication more straightforward. As for transmission scheduling, the system would adopt a carrier sense multiple access with collision avoidance (CSMA/CA) protocol to communicate with the other devices [31]. The device first listens to the UWB channel to check for transmitting nodes. If another node is detected, the device waits a random interval for that node to finish before checking again and then sending its data. This technique is known to be unreliable due to the hidden node problem; however, given UWB's low transmission rate, the UWB range packets being of small size, and the expected sparsity of the robot at a specific time (no more than ten robots at a time) the risk for package collision is low. The CSMA/CA would be able to handle such traffic with acceptable system delay [32]. If, after sending, packets are not returned within a time frame T, the ranging handshake would be sent again.

Following the data collection, several data processing checks were employed before following through with the initial localization step:

1. If the current robot was stationary, then a minimum of three unique anchors were required
2. If the current robot was non-stationary and the anchors were stationary, then a minimum of two anchors were required
3. If the current robot was non-stationary and the anchors were non-stationary, then a minimum of one anchor was required

The robots were determined to be mobile by looking at the reported robot's positions and ensuring the distances between points were above a certain tolerance, giving enough information for the non-linear least square algorithm to work.

To mitigate issues with sensor noise, a minimum number of data points was enforced to start the trilateration. Once the quota was fulfilled, the non-linear least square algorithm was performed as described in Section 2.4, the robot was defined as localized, and the refining process using the UKF was started. However, if the node failed to localize within a specific time frame, the robot relied solely on odometry for the localization process and "positioned itself within its relative positioning system." Suppose the node was at any time able to satisfy all the conditions to localize due to it coming in range with other localized nodes. In that case, the robot switched from using the relative positioning system to localizing itself in the new global reference frame and continued using the UKF to refine the results. At this point, the robot switched from an unlocalized node to a localized node. By constantly localizing and changing status once localized, the robots eventually converged to a network where all the nodes followed the same reference frame.

2.2. Parameter Definitions

A detailed definition of all parameters used in the system will be presented in this subsection. For convention, the vectors $\boldsymbol{p}$ and $\boldsymbol{\chi}$, $\boldsymbol{\phi}$ represented the global position, relative position, and heading of the node, respectively. The heading was defined where 0 radians points in the positive x-axis of $\boldsymbol{p}$, and

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

a counterclockwise motion represented an increase in the heading. χ represented the inter distance between two points in an agreed reference frame and can be formulated as,

$$\chi_{ij} = \overrightarrow{P_i P_j} = p_j - p_i = (x_j - x_i, y_j - y_i, z_j - z_i) \tag{1}$$

The relative Euclidean distance between two nodes was thus denoted as

$$\chi_{ij} = r_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2} \tag{2}$$

The environment consisted of a team of UAV, UGV, and static UWB anchors labeled 1,2,...N. The calculated global transformation matrices of the UGV were defined as the transformation from the center of the robot's relative origin point. The starting position of the odometry was assumed to be $(0, 0, 0)$ with 0 heading, linear and angular velocity, to the robots' global position $(\chi_{0,0})$. The transformation thus translated and rotated the static odometry frame of the robot. To solve the problem, each UGV_i was able to access the odometry data, denoted as $(\chi_{x_i,0}, \chi_{y_i,0}, \chi_{z_i,0})$ with angular velocity ω_i, linear velocity v_i, heading $\Delta\theta_i$ also be represented as $\delta\theta = \arctan\left(\frac{v_y}{v_x}\right)$, $\dot{\theta}$ represented the yaw rate. The robot had distance measurements to its neighbor UGV_j, i.e., $r_{ij} = \chi_{ij}$. It was assumed that the robot was moving on a flat 2D plane. As such, $\hat{p}_z$ remained constant. The global position p was then derived from $(x_0, y_0, z_0), \theta_0$ by:

$$p = \begin{bmatrix} \cos\theta_0 & -\sin\theta_0 & 0 & 0 & x_0 \\ \sin\theta_0 & \cos\theta_0 & 0 & 0 & y_0 \\ 0 & 0 & 1 & 0 & z_0 \\ 0 & 0 & 0 & 1 & \theta_0 \end{bmatrix} \begin{bmatrix} \chi_{x,i} \\ \chi_{y,i} \\ \chi_{z,i} \\ \Delta\theta_i \\ 1 \end{bmatrix} \tag{3}$$

Each UGV_i denoted the set of its neighbors as UGV_j where $\{j \mid j \in \mathbb{Z} \text{ and } i \neq j\}$. The neighbor transmitted its range measurements, and UWB anchor position, A_k. The position measurement data was located in the robot's global reference frame, i.e., $p_j = (x_j, y_j, z_j)$. Each UWB anchor was annotated as A_k with a unique identification k where $\{k \mid k \in \mathbb{Z}\}$, similarly each UWB tag was denoted as T_w with a unique identification w where $\{w \mid w \in \mathbb{Z}\}$.

2.3. Initial Localization

An initial position estimate was made on the target node. The proposed algorithm solved this problem by structuring the estimate as a non-linear least square (NLS) problem. To simplify the construction of the NLS, the odometry and neighboring range data of a single robot node were used to calculate $(x_0, y_0, z_0), \theta_0$. The position estimate was later refined using the UKF.

The geometry of the robot is defined and elaborated on below. There were two tags on the robot, which were separated by a fixed distance d. In the global reference frame, the tags were defined as T_w. In the static inertial reference frame of the robot, the position of the tags was represented as t_w where the left tag was at $t_l = (0, \frac{d}{2})$, and the right tag was at $t_r = (0, -\frac{d}{2})$. In addition, the odometry position was measured from the robot's defined origin, $(0, 0)$, which was the midpoint of the right and left tags. $p_i = \frac{T_{l,w1} + T_{r,w2}}{2}$ and therefore $(0, 0) = \frac{t_l + t_r}{2}$. The range recorded by the UWB tags was represented as $A_j - T_w = d_{jw}$, where A_j was a UWB anchor located on a UGV neighbor.

The robot's heading ϕ_i was considered when solving the non-linear least squares problem by compensating the target distance by rotating the tag pose. The relative tag position defined as $(T_w - p_i)$, was represented as the vector t_w rotated about the center of the robot (p_i) by the heading global heading ϕ_i.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

2.4. Non-Linear Least Squares

The non-linear least squares optimization method seeked to minimize the following:

$$S = \sum_{i=1}^{m} r_i^2 \tag{4}$$

where r_i was the residual given by:

$$r_i = d_{jw,calculated}^2 - d_{jw,measured}^2 \tag{5}$$

The range measurement d_{jw} was defined as a function of $\boldsymbol{p_i}, \phi_i, t_w$ and A_j. In addition, $\Delta\theta_i$ represented the current heading measured from the odometry and χ_i represented the odometry position $(\chi_{x,i}, \chi_{y,i}, \chi_{z,i})$:

$$d_{jw} = \vec{A}_j - \vec{T_w} \tag{6}$$

$$= \vec{A}_j - \left(\vec{\boldsymbol{p_i}} + \begin{bmatrix} \cos\phi_i & -\sin\phi_i & 0 \\ \sin\phi_i & \cos\phi_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{t}_j \right) \tag{7}$$

$$= \| \vec{A}_j - ((\vec{\boldsymbol{p_0}} + \begin{bmatrix} \cos\theta_0 & -\sin\theta_0 & 0 \\ \sin\theta_0 & \cos\theta_0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \chi_i) + \tag{8}$$

$$\begin{bmatrix} \cos(\theta_0 + \Delta\theta_i) & -\sin(\theta_0 + \Delta\theta_i) & 0 \\ \sin(\theta_0 + \Delta\theta_i) & \cos(\theta_0 + \Delta\theta_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{t_w}) \| \tag{9}$$

$$\begin{aligned} d_{jw}^2 = {} & (A_{jx} - (x_0 + \left[\chi_{x,i}\cos\theta_0 - \chi_{y,i}\sin\theta_0\right] + \\ & \left[t_{wx}\cos(\theta_0 + \Delta\theta_i) - t_{wy}\sin(\theta_0 + \Delta\theta_i)\right]))^2 + \\ & (A_{jy} - (y_0 + \left[\chi_{x,i}\sin\theta_0 + \chi_{y,i}\cos\theta_0\right] + \\ & \left[t_{wx}\sin(\theta_0 + \Delta\theta_i) + t_{wy}\cos(\theta_0 + \Delta\theta_i)\right]))^2 + \\ & \left(A_{jz} - (z_0 + t_{wz})\right)^2 \end{aligned} \tag{10}$$

The minimum value of S could be found when the gradient was 0. As a result, Jacobian was used to calculate the partial derivative of all the n variables needed.

$$\frac{\delta S}{\delta \beta_j} = 2\sum_{i=1}^{m} r_i \frac{\delta r_i}{\delta \beta_j} = 0 \qquad (j = 1, \cdots, n) \tag{11}$$

Due to the gradient equations not having a closed solution, the variables β_j were solved iteratively with Newton's iteration method.

$$f(x_i, \beta) \approx f(x_i, \beta^k) + \sum_j J_{ij} \Delta\beta_j \tag{12}$$

The Jacobian is a matrix of partial derivatives as a function of constants, the independent variable, and the parameter. It was constructed as such:

$$\frac{\delta r_i}{\delta \beta_j} = J_{ij} \tag{13}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

A single row of the Jacobian (J) was derived from the d_{iw}^2 function below. In this matrix c, s represent $\cos(\cdot)$ and $\sin(\cdot)$ respectively.

$$J^T = \begin{bmatrix} \frac{\delta d_{jw}^2}{\delta x_0} \\ \frac{\delta d_{jw}^2}{\delta y_0} \\ \frac{\delta d_{jw}^2}{\delta z_0} \\ \frac{\delta d_{jw}^2}{\delta \theta_0} \end{bmatrix} = \begin{bmatrix} -2(A_x - t_x c(\theta_0 + \Delta\theta_i) + t_y s(\theta_0 + \Delta\theta_i) \\ -c(\theta_0)\chi_{x,i} - x_0 + s(\theta_0)\chi_{y,i}) \\ \\ 2(-A_y + t_x s(\theta_0 + \Delta\theta_i) + t_y c(\theta_0 + \Delta\theta_i) \\ +s(\theta_0)\chi_{x,i} + c(\theta_0)\chi_{y,i} + y_0) \\ \\ 2(-A_z + t_z + z_0) \\ \\ 2(t_x c(\theta_0 + \Delta\theta_i) - t_y s(\theta_0 + \Delta\theta_i) + \\ c(\theta_0)\chi_{x,i} - s(\theta_0)\chi_{y,i}) \\ (-A_y + t_x s(\theta_0 + \Delta\theta_i) + t_y c(\theta_0 + \Delta\theta_i) + \\ s(\theta_0)\chi_{x,i} + c(\theta_0)\chi_{y,i} + y_0) \\ -2(t_x s(\theta_0 + \Delta\theta_i) + t_y c(\theta_0 + \Delta\theta_i) + \\ s(\theta_0)\chi_{x,i} + c(\theta_0)\chi_{y,i}) \\ (-A_x + t_x c(\theta_0 + \Delta\theta_i) - t_y s(\theta_0 + \Delta\theta_i) + \\ c(\theta_0)\chi_{x,i} + x_0 - s(\theta_0)\chi_{y,i}) \end{bmatrix}$$

Using Newton's iteration algorithm for minimizing the error, the Jacobian in equation (14) and the residual functions r_i were calculated at each step. The residual function was redefined as $f_i = d_{jw,i}(x_0, y_0, z_0, \theta_0)^2 - d_{jw,i,\text{measured}}^2$ and the vectors $\boldsymbol{f}$ and $\boldsymbol{\beta}$ were introduced as:

$$\boldsymbol{J} = 2\begin{bmatrix} \frac{\delta f_1}{\delta x_0} & \frac{\delta f_1}{\delta y_0} & \frac{\delta f_1}{\delta z_0} & \frac{\delta f_1}{\delta \theta_0} \\ \frac{\delta f_2}{\delta x_0} & \frac{\delta f_2}{\delta y_0} & \frac{\delta f_2}{\delta z_0} & \frac{\delta f_2}{\delta \theta_0} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\delta f_n}{\delta x_0} & \frac{\delta f_n}{\delta y_0} & \frac{\delta f_n}{\delta z_0} & \frac{\delta f_n}{\delta \theta_0} \end{bmatrix}, \boldsymbol{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{bmatrix}, \boldsymbol{\beta} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ \theta_0 \end{bmatrix} \tag{14}$$

Thus, the Newton iteration became:

$$\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} - \left(\boldsymbol{J_k^T J_k}\right)^{-1} \boldsymbol{J_k^T f_k} \tag{15}$$

where $\boldsymbol{\beta_{k+1}}$ was the new estimated parameter values and $\boldsymbol{\beta_k}$ was the last approximation. The initial estimate needed for the Newton approximation (β_0) was provided in the simulation. The parameter β_0 can be initialized in one of two wats: it can either be set to an initial state of 0, or it can be provided a coarse estimate of the robot's geographic area. The algorithm terminated when $\boldsymbol{\beta_{k+1}} - \boldsymbol{\beta_k} \leq \epsilon$, where ϵ was a defined termination tolerance criterion.

This was the approach to using the non-linear least square optimization, and many other optimizations could be implemented to solve this problem. The scipy.optimize.least_squares function was used in this work to implement the non-linear least square calculations.

2.4.1. Monte Carlo Simulation

A Monte Carlo simulation was developed to demonstrate the convergence characteristics of the initial localization process. This simulation involved the manipulation of two critical variables: the number of robots traversing the environment,'N,' and the time horizon for the odometry history, 'T.' In the simulation, 'N' random robots were created, each adhering to a differential drive motion model. At each time step (0.1s), these simulated robots moved with a random linear velocity selected from the range of [-vmax/2, vmax], favoring forward motion, and an angular velocity sampled from [-wmax,

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

wmax], both vmax and wmax were set to 2. Simultaneously, one robot was randomly selected to transmit its range measurements at each time step.

The parameter 'T' governed the duration before the localization process was conducted using the collected data. Random noise was introduced to the acquired odometry data, affecting both anchor poses and the target robot's measurements. The noise levels were standardized at 0.5 cm for the x and y coordinates of the odometry anchor poses, 2 cm for the range measurements, and 1/1000 radians for angular measurements. The target robot was randomly placed within a 10x10m world with a randomly assigned orientation as demonstrated in Figure 1. Additionally, the Non-Linear Least Squares' initial guesses were set to zero.

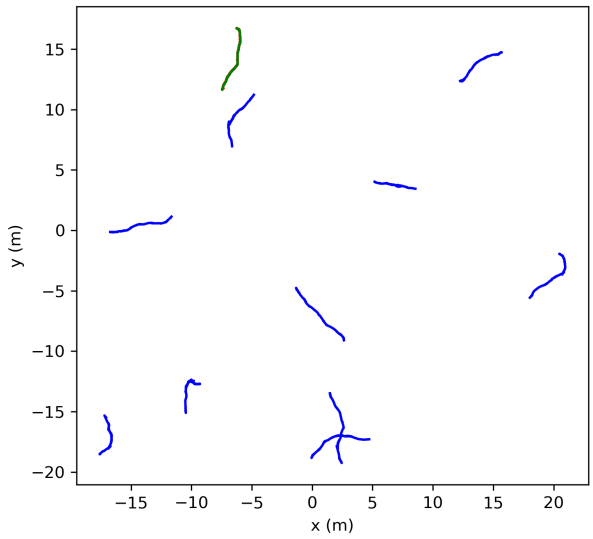

Figure 1. Monte Carlo simulation example with 'T'=10 and 'N'=10 (Green robot is the target robot).

Each parameter combination simulated 1000 samples to collect the aggregated information. These simulations generated values of 'T' from 1s to 10s and an 'N' ranging from 1 to 5 robots. The results, as depicted in Figures 2 and 3, indicate that as the time horizon 'T' increases, the Euclidean error relative to the target transformation matrix diminishes significantly. Furthermore, a higher number of robots present led to less variation in the error relative to the target across runs, and the system was less likely to reach an incorrect solution.

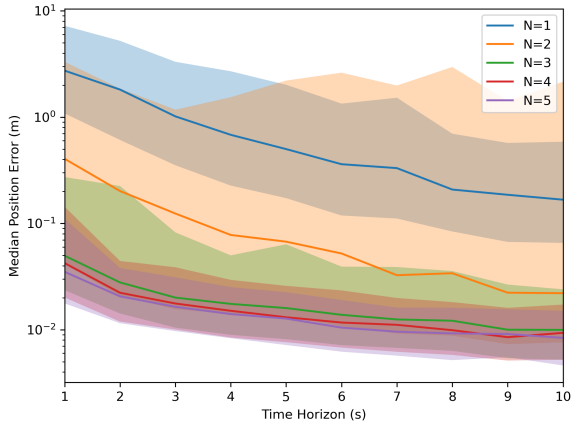

Figure 2. Monte Carlo simulation of the Euclidean distance error.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

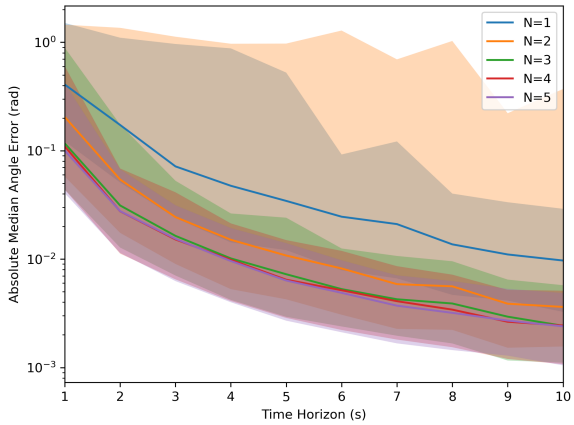

Figure 3. Monte Carlo simulation of the absolute angle error.

To demonstrate the robustness of the system relative to noise levels. The Monte Carlo simulations were generated again; however, this time with assuming a constant 'T' of 10 second and a varying scale of the noise levels listed above at different magnitudes.

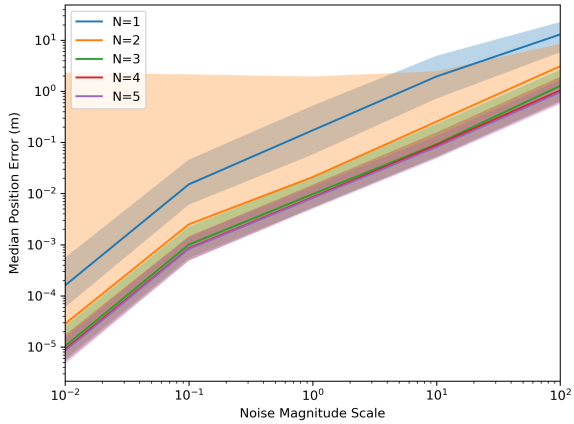

Figure 4. Monte Carlo of Euclidean distance error with varying noise scale.

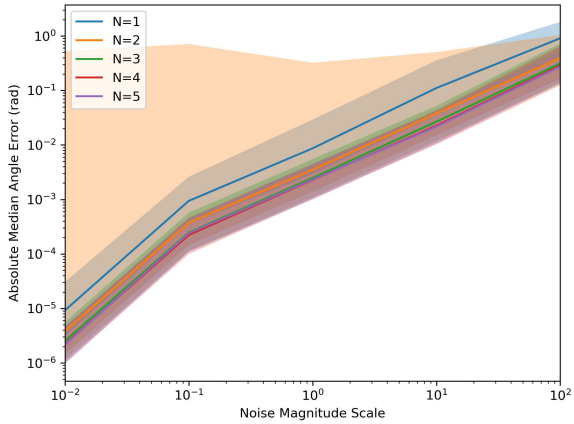

Figure 5. Monte Carlo of angle error with varying noise scale.

2.5. UKF Position Refinement

In the following sections, the robot has already been localized and given an initial position estimation. Further refinement of the results by reducing the sensor noise was achieved using a UKF

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

[33,34]. Two types of data were used in the UKF. The first came from the ranging measurements at time step k, d_k^{jw}, between the tag T_w on the target node and a localized neighbor anchor A_j. The second piece of data came from the robot's onboard odometry. The following sections go through the UKF model for the CTRV motion model.

2.5.1. Motion model

The UKF motion model that this article followed was the constant turn rate and velocity magnitude model (CTRV). This non-linear motion model assumed that the node could move straight but turned following a bicycle turn model with constant turn rate and linear velocity [35]. Other types of motion models were available, as illustrated in Figure 6. The constant velocity (CV) motion model was the simplest one available, a linear motion model where the linear acceleration was defined to be 0. The constant turn rate and acceleration (CTRA) is an expanded version of the CTRV motion model where the acceleration is accounted for and determined. Similarly to the CTRA motion model, the constant curvature and acceleration (CCA) model replaces the yaw rate of the model with the curvature instead. The constant steering angle and velocity (CSAV) model returns to having the acceleration constant and replacing the assumes constant steering angle instead. Each model has its own set of advantages and disadvantages; however, the CTRV motion model was chosen thanks to the balance in computation speed and accuracy in comparison with each different model [35]. The CTRV was also selected because the odometry sensor output contains the same state variables. This research problem formulation assumes the nodes of interest are ground robots; however, a general motion model could be used. As long as the robot's dynamics can be derived, the system's motion model can be used with this implementation. For drones, the motion model is linear. This eliminates the need for an unscented Kalman filter model, allowing the use of a linear Kalman filter instead.

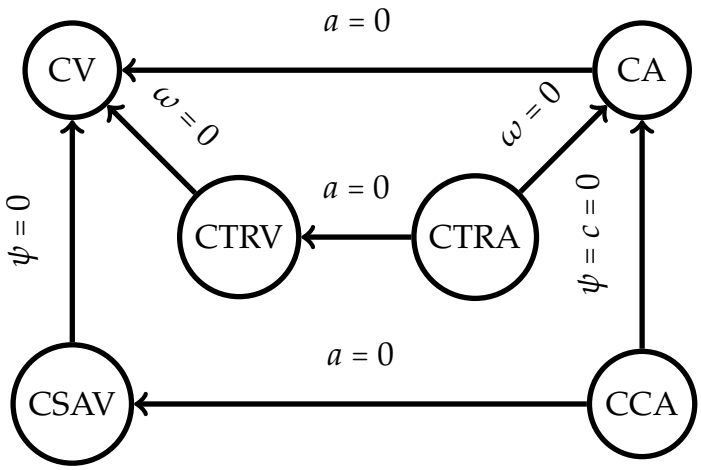

Figure 6. Motion Model hierarchy.

The state vector of the CTRV model was defined to be (Figure 7):

$$x = \begin{bmatrix} p_x & p_y & p_z & v & \psi & \dot{\psi} \end{bmatrix}^T \tag{16}$$

where v was the node's speed, ψ was the yaw angle which described the orientation according to Figure 7, and $\dot{\psi}$ represented the yaw rate. The change of rate in the state was expressed as:

$$\begin{aligned} \dot{x} &= \begin{bmatrix} \dot{p_x} & \dot{p_y} & \dot{p_z} & \dot{v} & \dot{\psi} & \ddot{\psi} \end{bmatrix}^T \\ &= \begin{bmatrix} v \cdot \cos\psi & v \cdot \sin\phi & 0 & 0 & \dot{\psi} & 0 \end{bmatrix}^T \end{aligned} \tag{17}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

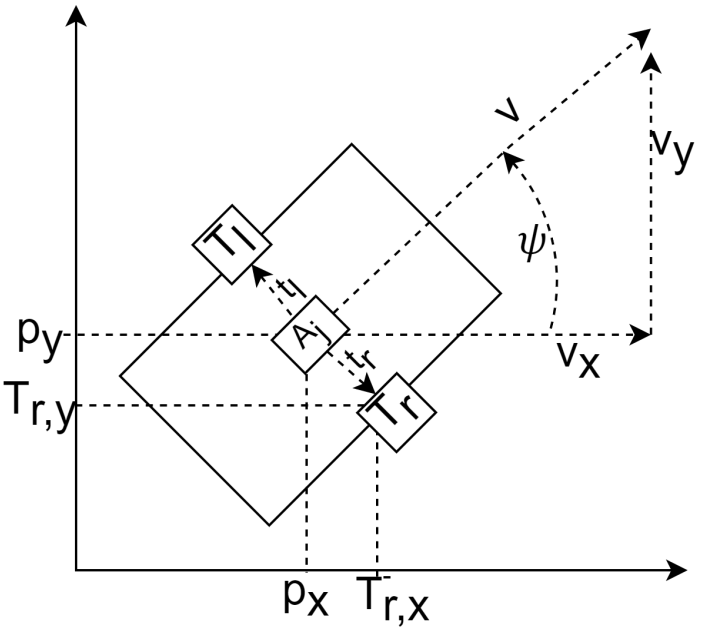

Figure 7. Mobile robot motion state.

The current step in the process was denoted by k, and the subsequent step by $k+1$. Consequently, the time difference was expressed as $\Delta t = t_{k+1} - t_k$. The process model, which predicted the state at $k+1$, could be decomposed into the deterministic and stochastic parts. To derive the deterministic part:

$$x_{k+1} = f_2(x_k) \tag{18}$$

$$= x_k + \int_{t_k}^{t_{k+1}} \left[v{\cdot}\cos\psi(t) \;\; v{\cdot}\sin\phi(t) \;\; 0 \;\; 0 \;\; \dot{\psi} \;\; 0 \right]^T dt \tag{19}$$

$$= x_k + \begin{bmatrix} v_k \int_{t_k}^{t_{k+1}} \cos\psi(t) dt \\ v_k \int_{t_k}^{t_{k+1}} \sin\phi(t) dt \\ 0 \\ 0 \\ \dot{\psi}\Delta t \\ 0 \end{bmatrix} \tag{20}$$

$$= x_k + \begin{bmatrix} v_k \int_{t_k}^{t_{k+1}} \cos\left(\psi_k + \dot{\phi}_k * (t - t_k)\right) dt \\ v_k \int_{t_k}^{t_{k+1}} \sin\left(\psi_k + \dot{\phi}_k * (t - t_k)\right) dt \\ 0 \\ 0 \\ \dot{\psi}\Delta t \\ 0 \end{bmatrix} \tag{21}$$

$$= x_k + \begin{bmatrix} \frac{v_k}{\dot{\psi}_k} \left(\sin\left(\psi_k + \dot{\psi}_k \Delta t\right) - \sin\psi_k\right) \\ \frac{v_k}{\dot{\psi}_k} \left(-\cos\left(\psi_k + \dot{\psi}_k \Delta t\right) + \cos\psi_k\right) \\ 0 \\ 0 \\ \dot{\psi}\Delta t \\ 0 \end{bmatrix} \tag{22}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

However, problems arose when $\dot{\psi} \approx 0$ as this would cause a division by 0. A modified version of the motion model was thus defined for when $\dot{\psi} \leq \varepsilon$:

$$f_2(x_k) = x_k + \begin{bmatrix} v_k \cos\psi_k \Delta t & v_k \sin\psi_k \Delta t & 0 & 0 & \dot{\psi}\Delta t & 0 \end{bmatrix}^T \tag{23}$$

Next, the stochastic component was designated as the noise vector, encompassing the linear and yaw acceleration noises in a CTRV model. At time step k, the noise ν was characterized as follows:

$$\nu_k = \begin{bmatrix} \nu_{a,k} & \nu_{\ddot{\psi},k} \end{bmatrix}^T \tag{24}$$

where $\nu_{a,k}$ was the linear acceleration noise defined as a normal distribution, $\nu_{a,k} \sim \mathcal{N}\left(0, \sigma_a^2\right)$ and $\nu_{\ddot{\psi},k}$ was the yaw acceleration noise defined as a normal distribution, $\nu_{\ddot{\psi},k} \sim \mathcal{N}\left(0, \sigma_{\ddot{\psi}}^2\right)$. It was assumed that the linear and angular acceleration would remain relatively constant during small time intervals, resulting in approximately linear motion between two timesteps (this assumption was valid unless the yaw rate was excessively high). As a result, the noise function was expressed as follows:

$$f_2(\nu_k) = \begin{bmatrix} \frac{1}{2}(\Delta t)^2 \cos\psi_k \cdot \nu_{a,k} \\ \frac{1}{2}(\Delta t)^2 \sin\psi_k \cdot \nu_{a,k} \\ 0 \\ \Delta t \cdot \nu_{a,k} \\ \frac{1}{2}(\Delta t)^2 \cdot \nu_{\ddot{\psi},k} \\ \Delta t \cdot \nu_{\ddot{\psi},k} \end{bmatrix} \tag{25}$$

The full motion model was characterized as follows:

$$f(x_k, \nu_k) = f_1(x_k) + f_2(\nu_k) \tag{26}$$

2.5.2. State Prediction

- Sigma Points

The first step to the unscented Kalman filter was the sigma point generation. Using sigma points is the main difference between the EKF and the UKF. The EKF linearizes the system through a Taylor-series expansion around the mean of the relevant Gaussian random variable (RV) [34]. Using multiple points to sample the state distribution improved linearization of the non-linear space [34]. When greater accuracy was required, a UKF was recommended compared to an EKF. Due to the addition and the linear scaling of the number of sigma points required based on the dimensionality of the state vector, UKFs are known to be more computationally expensive. Following a Gaussian distribution, the sigma points were generated from the last updated state and covariance matrix.

First, the augmented state and covariance matrix were formulated, with the normal state vector denoted by:

$$x_k = \begin{bmatrix} p_{x,k} & p_{y,k} & p_{z,k} & v_k & \psi_k & \dot{\psi}_k \end{bmatrix}^T \tag{27}$$

Having a dimension of $n_x = 6$, the covariance matrix $P_{k|k}$ took the form of an $n_x \times n_x$ matrix. Meanwhile, the noise vector was characterized as:

$$\nu_k = \begin{bmatrix} \nu_{a,k} & \nu_{\ddot{\psi},k} \end{bmatrix}^T \tag{28}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

with dimension $n_\nu = 2$, the augmented state matrix was represented as follows:

$$x_{aug} = \left[x|\nu_k\right]^T \tag{29}$$

$$= \left[\, p_{x,k} \;\; p_{y,k} \;\; p_{z,k} \;\; v_k \;\; \psi_k \;\; \dot{\psi}_k \;\; \nu_{a,k} \;\; \nu_{\ddot{\psi},k} \,\right]^T \tag{30}$$

The augmented state dimension, determined as $n_a = n_x + n_\nu$, yielded a total of eight dimensions for the CTRV model, specifically $n_a = 8$. Subsequently, the process noise covariance matrix was constructed, incorporating the expected acceleration and yaw rate Gaussian distribution into the matrix Q, resulting in the following representation:

$$Q = E\{\nu_k \cdot \nu_k^T\} = \text{diag}\left(\begin{bmatrix}\sigma_a^2 & \sigma_{\ddot{\psi}}^2\end{bmatrix}\right) \tag{31}$$

The augmented covariance matrix was thus represented as:

$$P_{a,k|k} = \text{diag}\left(\begin{bmatrix}P_{k|k} & Q\end{bmatrix}\right) \tag{32}$$

The augmented covariance matrix became a square matrix with size $n_a \times n_a = 8 \times 8$. By convention, it was recommended to have at least $n_\sigma = 2n_a + 1$ sigma points; in the case of the illustrated model, this would be $n_\sigma = 2(8) + 1 = 17$. The sigma points were then calculated as a list

$$X_{k|k} = \left[\, x_{k|k} \;\; x_{k|k} + \sqrt{(\lambda + n_a)P_{k|k}} \;\; x_{k|k} - \sqrt{(\lambda + n_a)P_{k|k}} \,\right] \tag{33}$$

where λ was the scaling factor defined as $\lambda = \alpha^2\left(n_x + 3 - n_a\right) - n_x$, where λ dictated how far away from the mean the sigma points would be positioned [36]. The parameter α defined the spread of the sigma points around x_k and was set to a small positive value between 0 and 1. By default, it was set to 1; however, this could be tuned if necessary. When taking $\sqrt{P_{k|k}}$ for the sigma points, there were multiple ways to take the square root of a matrix; however, Cholesky decomposition was recommended due to its computation speed. The output of this list should be in the format $n_a \times n_\sigma$.

- Prediction Step

Once the sigma points were calculated and transformed into their respective predicted state, each was transformed using the process model and then saved to another list now of format $n_x \times n_\sigma$.The processed state sigma points were in the format $x_{k+1|k} = \begin{bmatrix}p_{x,k+1} & p_{y,k+1} & p_{z,k+1} & v_{k+1} & \psi_{k+1} & \dot{\psi}_{k+1}\end{bmatrix}^T$
$X_{a,k|k}$, the sigma points of the augmented $f(x_k, \nu_k)$, the process model function $X_{k+1|k} = \emptyset\; x_{a,k|k,i} \in X_{a,k|k}\; x_{x,i} \leftarrow x_{a,k|k,i}[0 : n_x]\; x_{\nu,i} \leftarrow x_{a,k|k,i}[n_x : n_a]\; x_{k+1|k,i} \leftarrow f(x_{x,i}, x_{\nu,i})$ Append $x_{k+1|k,i}$ to $X_{k+1|k}$
$X_{k+1|k}$

- Calculate Mean and Covariance from the Predicted Points

In this step, the predicted state mean vector and the predicted covariance matrix were calculated by aggregating the sigma points using a weighted average of the points. There were multiple ways to initialize the weights for this part; however, this paper sticks to a standard Gaussian distribution.

$$\omega_i^m = \frac{\lambda}{(\lambda + n_a)} \qquad i = 0 \tag{34}$$

$$\omega_i^c = \frac{\lambda}{(\lambda + n_a)} + (1 - \alpha^2 + \beta) \qquad i = 0 \tag{35}$$

$$\omega_i = \omega_i^m = \omega_i^c = \frac{1}{2\left(\lambda + n_a\right)} \qquad i = 1, \cdots, 2 * n_\sigma \tag{36}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

where λ and α were the same from the sigma point calculation 2.5.2. β was an extra parameter used to incorporate any prior knowledge of the distribution of the state. In most cases, this paper's β was left to be 0. Once the weights were calculated, the predicted mean and covariance were calculated respectively:

$$x_{k+1|k} = \sum_{i=0}^{2n_\sigma} w_i^m X_{k+1|k,i} \tag{37}$$

$$P_{k+1|k} = \sum_{i=0}^{2n_\sigma} w_i^c (X_{k+1|k,i} - x_{k_k+1|k})(X_{k+1|k,i} - x_{k_k+1|k})^T \tag{38}$$

2.5.3. Measurement Prediction

The measurement stage processes the sigma points into predicted measurement outputs. To accomplish this, a measurement function $h(x_{k+1})$ was developed for each of the sensor types, where x_{k+1} was the output of the process model in the prediction step 2.5.2. The measurement vector was z_{k+1} and ω_{k+1} the inherent measurement noise.

$$x_{k+1} = f(x_k, \nu_k) \tag{39}$$

$$z_{k+1} = h(x_{k+1}) + \omega_{k+1} \tag{40}$$

Due to this paper tackling the fusion of two sensor types, the odometry data and the UWB range data, two different $h(x_{k+1})$ functions needed to be defined. For the odometry data, the state vector was defined as:

$$z_{k+1|k} = h(x_{k+1}) = \begin{bmatrix} p_{x,k+1} & p_{y,k+1} & p_{z,k+1} & v_{k+1} & \psi_{k+1} & \dot{\psi}_{k+1} \end{bmatrix}^T \tag{41}$$

The measurement transformation function for the UWB ranging sensors took a similar format to the non-linear least square d_{jw} derivation (2.4) where the predicted UWB range measurement was derived given the estimated sigma point state (x_{k+1}), the known anchor point position (A_j) and the tag position relative to the node's reference frame (t_w).

$$\begin{aligned} z_{k+1|k} &= h\left(x_{k+1}, A_j, t_w\right) \\ &= \vec{A}_j - \vec{T_w} \\ &= \vec{A}_j - \left(\vec{p_{k+1}} + \begin{bmatrix} \cos\psi_{k+1} & -\sin\psi_{k+1} & 0 \\ \sin\psi_{k+1} & \cos\psi_{k+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \vec{t_j} \right) \\ &= \sqrt{\begin{array}{r} \left(A_{x,j} - \left(p_{x,k+1} + \left[t_{x,w}\cos\psi_{k+1} - t_{y,w}\sin\psi_{k+1}\right]\right)\right)^2 + \\ \left(A_{y,j} - \left(p_{y,k+1} + \left[t_{x,w}\sin\psi_{k+1} + t_{y,w}\cos\psi_{k+1}\right]\right)\right)^2 + \\ \left(A_{z,j} - \left(p_{z,k+1} + t_{z,w}\right)\right)^2 \end{array}} \\ &= \left[d_{jw}\right] \end{aligned} \tag{42}$$

The set of measurement sigma points was denoted as $\mathcal{Z}_{k+1|k}$. If the measurement used odometry, the output shape should be $n_x n_\alpha$, and if it was UWB range data, the output should be $1n_\alpha$.

Once the measurement sigma points were collected, the weighted predicted measurement means ($z_{k+1|k}$), and the predicted measurement covariance ($S_{k+1|k}$) were calculated, similar to how the mean and the covariance from the predicted state sigma points were calculated. The weights were defined in Section 2.5.2.

$$z_{k+1|k} = \sum_{i=0}^{2n_\sigma} \omega_i^m \mathcal{Z}_{k+1|k,i} \tag{43}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

The measurement noise covariance was also defined $R = E\{w_k \cdot w_k^T\}$ where the vector w_k was the measurement noise for the sensor's state. For the odometry sensor, R would be defined as:

$$R = \text{diag}\left(\begin{bmatrix}\sigma_x^2 & \sigma_y^2 & \sigma_z^2 & \sigma_v^2 & \sigma_v^2 & \sigma_\psi^2 & \sigma_\psi^2\end{bmatrix}\right) \tag{44}$$

While the R matrix for the UWB range sensor would be defined as:

$$R = \begin{bmatrix}\sigma_d^2\end{bmatrix} \tag{45}$$

Using the defined R matrices, the predicted measurement covariance could be defined.

$$S_{k+1|k} = \sum_{i=0}^{2n_\sigma} \omega_i^c \left(\mathcal{Z}_{k+1|k,i} - z_{k+1|k}\right)\left(\mathcal{Z}_{k+1|k,i} - z_{k+1|k}\right)^T + R \tag{46}$$

2.5.4. State Update

The actual state vector could be updated after defining the predicted state and measurement. First, the cross-correlation between the state points in state space and the measurement space ($T_{k+1|k}$) had to be computed:

$$T_{k+1|k} = \sum_{i=0}^{2n_\sigma} \omega_i^c \left(X_{k+1|k,i} - x_{k+1|k}\right)\left(\mathcal{Z}_{k+1|k,i} - z_{k+1|k}\right) \tag{47}$$

From there, the Kalman gain was calculated:

$$K_{k+1|k} = T_{k+1|k} S_{k+1|k}^{-1} \tag{48}$$

The current state and covariance matrix was then updated respectively, where z_{k+1} was the currently retrieved measurement data:

$$x_{k+1|k+1} = x_{k+1|k} + K_{k+1|k}\left(z_{k+1} - z_{k+1|k}\right) \tag{49}$$

$$P_{k+1|k+1} = P_{k+1|k} - K_{k+1|k} S_{k+1|k} K_{k+1|k}^T \tag{50}$$

3. Simulation Results

ROS Melodic using Gazebo 9 was used to simulate the described environment and used a set of 3 objects/robots to model the world. The first object was a simple static UWB beacon represented by a solid cylinder meant to represent a generic UWB anchor such as a tripod-mounted Decawave DWM1000 Module, this object would be given an agreed-upon location in global coordinates such as GPS, and as such, the estimated position would be static and with high precision. Decawave claims they can support up to 6.8 Mbps data rates with frequencies of 3.5GHz-6.5GHz, a centimeter-level accuracy, and maintain a connection for up to a range of 290 meters.

The second object was a Hector quadrotor, in Figure 8, with a UWB anchor mounted on the bottom of the drone [37]. This drone was meant to represent a moving UWB anchor. The drone would also be assumed to have been localized using GPS or cameras to represent a high-precision localization estimate. Instead of GPS, drones could use feature positioning based on public geographical information to remove the need to utilize GPS and generate high-accuracy localization [38,39]. In addition, as previously demonstrated in the Monte Carlo simulations in Section 2.3 in Figures 4 and 5, even when noise is present in the system if the time horizon is sufficiently large, the system will be able to derive it's initial position parameters relatively accurately. Thus, even if the "localized nodes" contain noise information, the system will be able to provide an initial estimate of the position.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

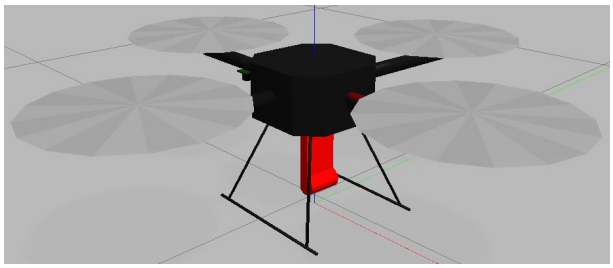

Figure 8. Hector drone.

The final object was the implementation of the Clearpath Jackals robots depicted in Figures 9 and 10 with two UWB tags on the top of its hood, equally spaced from the center axis of the robot with an extra centrally located UWB anchor. The tags allowed the robot to position itself, while the anchors allowed information propagation to help other robots localize themselves. In this study, it was assumed that the initial position and orientation of the Jackals were not provided. However, an initial estimate could be provided in the initial localization step to improve and reduce the initial position estimate error and thus improve the accuracy. As such, the localization of the Jackal was the main focus of the presented work.

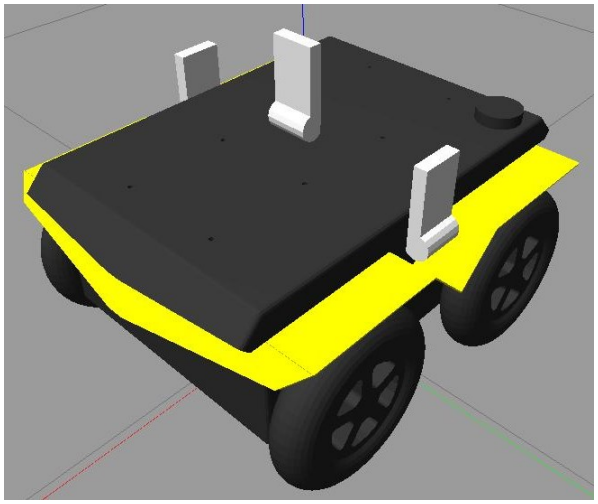

Figure 9. Simulated Jackal robot with UWB tags (left and right) and an anchor (middle).

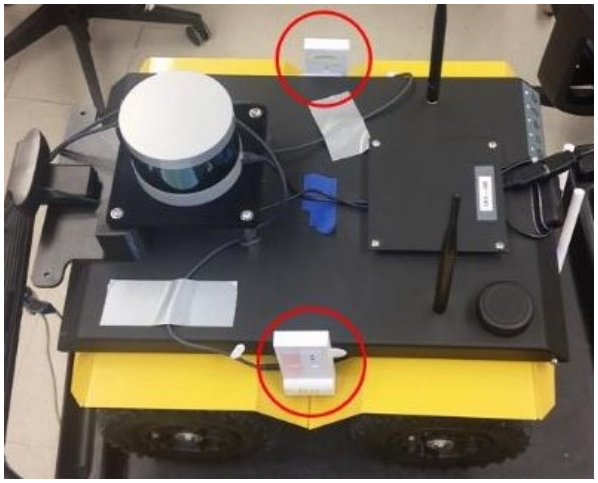

Figure 10. Jackal ground robot with DWMs mounted on left and right side [40].

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

The world consisted of two Jackals and three drones to simulate a mobile surveying endeavor, as presented in Figures 11 and 12. The drones represented the mobile anchors as they flew over the trees and had GPS to help localize themselves with acceptable accuracy. The drones also helped the ground Jackals, who did not have GPS. The process proceeded as such: initially, the drones flew over the trees to achieve GPS localization in a global reference frame; once localized, the Jackals developed their initial position estimation based on the Hector drones. Once the Jackals had localized themselves, they performed their abstract task. Once the task was complete, the drones moved to another location to be surveyed, and the Jackals followed suit. When the Jackals moved to the new location, the drones were too far or obstructed to send UWB ranging information, so to continue the localization, the two Jackals sent their inter-robot range measurements as well using their onboard odometry. Once the Jackals approached the new survey site, the UWB range measurements from the Hector drones helped correct any deviation during the offline path. The simulated noise measurements were based on the empirical data collected from the DWM1001-DEV in range increments of 1 meter up to 141 meters. The standard deviations of the measurements were recorded. The measurements also included scenarios when the UWB sensor was occluded by thin objects (< 25 cm) and thick objects (≥ 25 cm). The distributions can be found in the Gazebo UWB Plugin https://github.com/AUVSL/UWB-Gazebo-Plugin/blob/master/src/UwbPlugin.cpp.

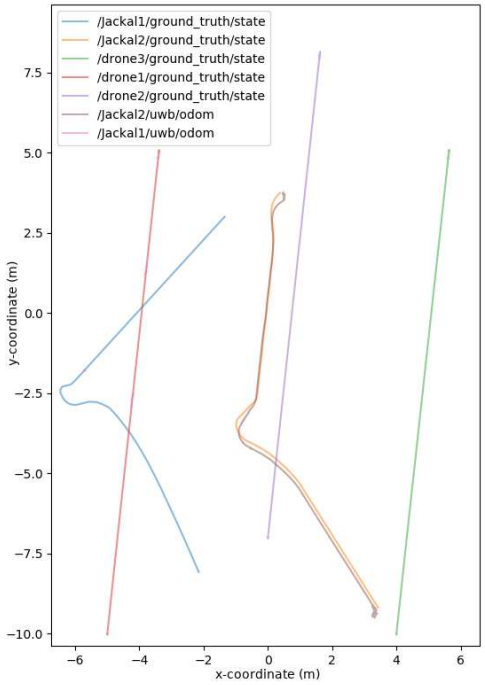

Figure 11. Jackal and Drone position plotting

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

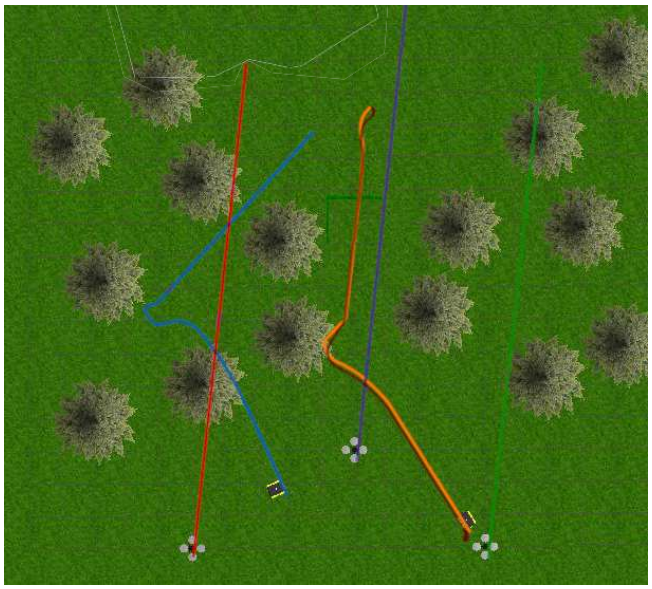

Figure 12. Gazebo environment position overlay.

Table 1. Drone and Mobile Jackal Simulation Results.

Gazebo worlds			
		Metrics	
RMSE x	0.0867 m	MAE x	0.2812 m
RMSE y	0.1092 m	MAE y	0.3074 m
RMSE z	0.0784 m	MAE z	0.2789 m
RMSE v	0.250m/s	MAE v	0.348m/s
RMSE ψ	0.035 rad	MAE ψ	0.150 rad
RMSE $\dot{\psi}$	0.66rad/s	MAE $\dot{\psi}$	0.81rad/s
RMSE pose	0.0256 m	MAE pose	0.1562 m

Two metrics were used to determine the accuracy of the unscented Kalman filter: the RMSE (Root Mean Squared error) and Mean Absolute Error (MAE). As demonstrated through the graphs, in Figures 13, 14 and 15, the unscented Kalman filter developed for the Jackals follows the ground truth very closely with minimal RMSE values of no greater than 0.10921 m for the positioning errors; however, it could be noted that the unscented Kalman filter had a more significant difficulty estimating the velocity and the yaw rate, with larger errors of 0.24959 $\frac{m}{s}$ and 0.65856 $\frac{\text{rad}}{s}$, though due to the difference in scales of the measurement values, this error was deemed to be acceptable. The MAE measurement values were relatively consistent, hovering around between 0.27 $\frac{m}{s}$ and 0.3 $\frac{\text{rad}}{s}$, which informs of a possible issue of systematic bias introduced in the system where the position of the system was consistently offset from the actual target. As noticed through the x, Figure 13, and y, Figure 14, position graphs, if noticed closely, the estimated values for the x and y position, when the values seem steady, tend to lag below the ground-truth value. Further tuning of the unscented Kalman filter model's parameters, such as the sensor noise matrix R, and the sigma point generation parameters α and β, could increase accuracy in the model. The MAE pose is the average position accuracy, a measure of how close, on average, the position of the robot was to the actual position of the robot using Euclidean distances $\left(\sqrt{(x_{\text{exp}} - x_{\text{act}})^2 + (y_{\text{exp}} - y_{\text{act}})^2 + (z_{\text{exp}} - z_{\text{act}})^2}\right)$. Thus, on average, the robot was about 15 cm away from its target position, as noticed by the RMSE, which was often with errors that had lower magnitudes.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 9 January 2024 doi:10.20944/preprints202401.0684.v1

Current literature does not provide a good reference comparison to this system. This is due to systems assuming a static environment or a static leader. However, there is no system where all anchors can move at once and still be localized.

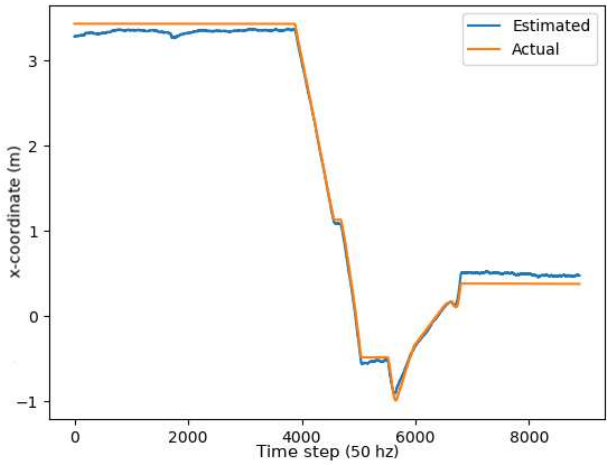

Figure 13. X Position plot.

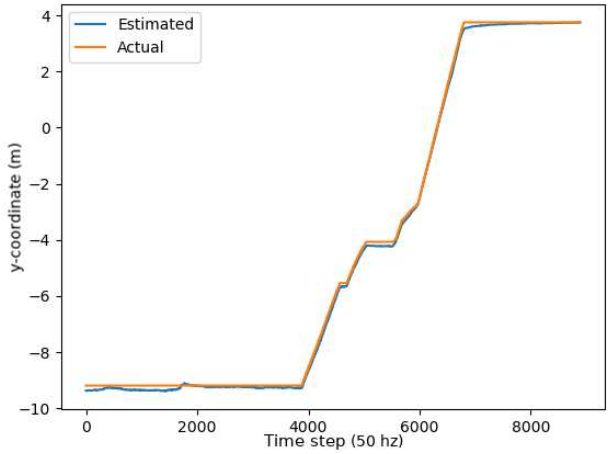

Figure 14. Y Position plot.

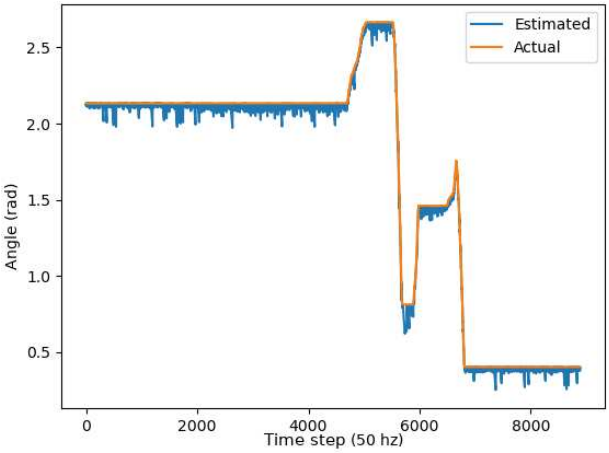

Figure 15. ψ plot.

 doi:10.20944/preprints202401.0684.v1

4. Conclusion and Future Work

This article discussed UWB localization using both stationary and mobile nodes. Specifically, it introduced an Ad-Hoc method to implement localization through a modified version of the AHLoS system. In this system, each node starts unlocalized or localized, and the localized nodes serve as reference points to the unlocalized nodes. Once there was enough information, the unlocalized nodes could determine a translation-rotation matrix from the relative position to the global reference frame. This was done using a non-linear least squares function which also accounts for the tag sensor's offset range measurements. The second phase of the paper transitioned to the UKF-based measurement data refinement stage, where both the globally translated odometry and the offset range measurements were fed into a CTRV motion-based UKF model to improve the long-term accuracy of the positioning system. Simulations were conducted to help provide proof of validity to the algorithms.

The current system assumes a 2D setup for the robot, which would cause inaccuracies in an inclined outdoor situation as ranging estimation for the robot's z value would not be correctly estimated, as the robot's 3D dynamics would not be reflected. Another problem is that even though the possibility of getting an inaccurate initial position estimate decreases over time, there are no guarantees that the system will converge with this current system. More robust optimization systems will be explored in the future with explicit guarantees. In addition, the system still requires fine-tuning of specific constants such as the initial P, Q, and R matrics of the UKF, and the time horizon 'T' and the number of robots N for the initial localization, as suboptimal performance would be achieved if the system is not correctly estimated.

In the future, the UKF will extend the account for the robot's yaw, pitch, and roll to better account for the diverse terrain and not assume that it was on flat terrain. To further improve the accuracy, additional sensors such as cameras and land-based markers to get a better position estimate and optical flow sensors to improve the velocity calculations will help further improve the UKF results. In addition, the development of the initial position algorithm to account for a wider variety of scenarios and improve the overall accuracy will be explored. Furthermore, a more robust adaptive Kalman filter approach for UWB sensors, to account for multipath and non-line-of-sight (NLOS) measurement error, will be explored to account for a more extensive range of terrain and reduce overall position error.

5. Installation

The code for this repository, for the simulated Gazebo environment, can be found https://github.com/AUVSL/UWB-Jackal-World and the code for the UKF can be found https://github.com/AUVSL/UWB-Localization. Both repositories contain READMEs with instructions to install/run the programs/environments. Annotated video demonstrations of the algorithms in question are located here: https://youtu.be/UbNkR3y-_S0 and https://youtu.be/S6px8JHvk-I

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Lu, W.; Rodríguez F., S.A.; Seignez, E.; Reynaud, R. Lane Marking-Based Vehicle Localization Using Low-Cost GPS and Open Source Map. *Unmanned Systems* **2015**, *03*, 239–251. https://doi.org/10.1142/S2301385015400014.
2. Zhao, L.; Wang, D.; Huang, B.; Xie, L. Distributed Filtering-Based Autonomous Navigation System of UAV. *Unmanned Systems* **2014**, *03*, 17–34. https://doi.org/10.1142/S2301385015500028.
3. Miller, M.; Soloviev, A.; Haag, M.; Veth, M.; Raquet, J.; Klausutis, T.; Touma, J. Navigation in GPS Denied Environments: Feature-Aided Inertial Systems **2011**. *116*, 7–8.
4. Hussein, H.H.; Radwan, M.H.; El-Kader, S.M.A. Proposed Localization Scenario for Autonomous Vehicles in GPS Denied Environment BT - Proceedings of the International Conference on Advanced Intelligent Systems and Informatics 2020, Cham, 2021; pp. 608–617.

5. Alarifi, A.; Al-Salman, A.; Alsaleh, M.; Alnafessah, A.; Al-Hadhrami, S.; Al-Ammar, M.A.; Al-Khalifa, H.S. Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances. *Sensors (Basel, Switzerland)* **2016**, *16*, 707. https://doi.org/10.3390/s16050707.
6. Geng, S.; Ranvier, S.; Zhao, X.; Kivinen, J.; Vainikainen, P. Multipath propagation characterization of ultra-wide band indoor radio channels. In Proceedings of the 2005 IEEE International Conference on Ultra-Wideband, 2005, pp. 11–15. https://doi.org/10.1109/ICU.2005.1569948.
7. Schejbal, V.; Cermak, D.; Nemec, Z.; Bezousek, P.; Fiser, O. Multipath Propagation Effects of UWB Radars. In Proceedings of the 2006 International Conference on Microwaves, Radar Wireless Communications, 2006, pp. 1188–1191. https://doi.org/10.1109/MIKON.2006.4345400.
8. Van Herbruggen, B.; Jooris, B.; Rossey, J.; Ridolfi, M.; Macoir, N.; Van den Brande, Q.; Lemey, S.; De Poorter, E. Wi-PoS: A Low-Cost, Open Source Ultra-Wideband (UWB) Hardware Platform with Long Range Sub-GHz Backbone. *Sensors (Basel, Switzerland)* **2019**, *19*, 1548. https://doi.org/10.3390/s19071548.
9. Thomä, R.; Hirsch, O.; Sachs, J.; Zetik, R. UWB Sensor Networks for Position Location and Imaging of Objects and Environments. 2007, Vol. 2007, pp. 1–9. https://doi.org/10.1049/ic.2007.1336.
10. García Núnez, E.; Poudereux, P.; Hernández, .; Ureña, J.; Gualda, D. *A robust UWB indoor positioning system for highly complex environments*; 2015; pp. 3386–3391. https://doi.org/10.1109/ICIT.2015.7125601.
11. Guo, K.; Li, X.; Xie, L. Ultra-Wideband and Odometry-Based Cooperative Relative Localization With Application to Multi-UAV Formation Control. *IEEE Transactions on Cybernetics* **2020**, *50*, 2590–2603. https://doi.org/10.1109/TCYB.2019.2905570.
12. Guo, K.; Qiu, Z.; Meng, W.; Xie, L.; Teo, R. Ultra-wideband based cooperative relative localization algorithm and experiments for multiple unmanned aerial vehicles in GPS denied environments. *International Journal of Micro Air Vehicles* **2017**, *9*, 169–186. https://doi.org/10.1177/1756829317695564.
13. Lensund, F.; Sjöstedt, M. Local positioning system for mobile robots using ultra wide-band technology. Student thesis, 2018.
14. Liu, J.; Pu, J.; Sun, L.; He, Z. An Approach to Robust INS/UWB Integrated Positioning for Autonomous Indoor Mobile Robots. *Sensors (Basel, Switzerland)* **2019**, *19*, 950. https://doi.org/10.3390/s19040950.
15. Nguyen, T.; Zaini, A.H.; Wang, C.; Guo, K.; Xie, L. Robust Target-Relative Localization with Ultra-Wideband Ranging and Communication. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), 2018, pp. 2312–2319. https://doi.org/10.1109/ICRA.2018.8460844.
16. Silva, O.D.; Mann, G.K.I.; Gosine, R.G. Development of a relative localization scheme for ground-aerial multi-robot systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 870–875. https://doi.org/10.1109/IROS.2012.6386015.
17. Yu, X.; Li, Q.; Queralta, J.P.; Heikkonen, J.; Westerlund, T. Cooperative UWB-Based Localization for Outdoors Positioning and Navigation of UAVs aided by Ground Robots. *ArXiv* **2021**, *abs/2104.00302*.
18. Güler, S.; Abdelkader, M.; Shamma, J.S. Peer-to-Peer Relative Localization of Aerial Robots With Ultrawideband Sensors. *IEEE Transactions on Control Systems Technology* **2020**, pp. 1–16. https://doi.org/10.1109/TCST.2020.3027627.
19. Zaeemzadeh, A.; Joneidi, M.; Shahrasbi, B.; Rahnavard, N. Robust Target Localization Based on Squared Range Iterative Reweighted Least Squares. *Proceedings - 14th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2017* **2017**, pp. 380–388. https://doi.org/10.1109/MASS.2017.50.
20. Beck, A.; Stoica, P.; Li, J. Exact and approximate solutions of source localization problems. *IEEE Transactions on Signal Processing* **2008**, *56*, 1770–1778. https://doi.org/10.1109/TSP.2007.909342.
21. Yin, F.; Fritsche, C.; Gustafsson, F.; Zoubir, A. TOA-based robust wireless geolocation and Cramér-Rao lower bound analysis in harsh LOS/NLOS environments. *IEEE Transactions on Signal Processing* **2013**, *61*, 2243–2255. https://doi.org/10.1109/TSP.2013.2251341.
22. Chartrand, R.; Yin, W. Iteratively reweighted algorithms for compressive sensing. *2008 IEEE International Conference on Acoustics, Speech and Signal Processing* **2008**, pp. 3869–3872. https://doi.org/10.1109/ICASSP.2008.4518498.
23. Gao, K.; Zhu, J.; Xu, Z. Majorization-Minimization-Based Target Localization Problem from Range Measurements. *IEEE Communications Letters* **2020**, *24*, 558–562. https://doi.org/10.1109/LCOMM.2019.2963834.

24. Kim, E.; Kim, K. Distance estimation with weighted least squares for mobile beacon-based localization in wireless sensor networks. *IEEE Signal Processing Letters* **2010**, *17*, 559–562. https://doi.org/10.1109/LSP.2010.2047169.
25. Sichitiu, M. Localization of wireless sensor networks with a mobile beacon. 10 2004, pp. 174–183. https://doi.org/10.1109/MAHSS.2004.1392104.
26. Fan, Q.; Sun, B.; Sun, Y.; Wu, Y.; Zhuang, X. Data Fusion for Indoor Mobile Robot Positioning Based on Tightly Coupled INS/UWB. *Journal of Navigation* **2017**, *70*, 1079–1097. https://doi.org/10.1017/S0373463317000194.
27. Harrison, R.L. Introduction To Monte Carlo Simulation. *AIP conference proceedings* **2010**, *1204*, 17–21. https://doi.org/10.1063/1.3295638.
28. Yoo, J.; Kim, W.; Kim, H.J. Distributed estimation using online semi-supervised particle filter for mobile sensor networks. *IET Control Theory Applications* **2015**, *9*, 418–427. https://doi.org/10.1049/IET-CTA.2014.0495.
29. Praveen Kumar, D.; Amgoth, T.; Annavarapu, C.S.R. Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion* **2019**, *49*, 1–25. https://doi.org/10.1016/J.INFFUS.2018.09.013.
30. Savvides, A.; Han, C.C.; Strivastava, M.B. Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors. In Proceedings of the Proceedings of the 7th Annual International Conference on Mobile Computing and Networking, New York, NY, USA, 2001; MobiCom '01, pp. 166–179. https://doi.org/10.1145/381677.381693.
31. Shenoy, N.; Hamilton, J.; Kwasinski, A.; Xiong, K. An improved IEEE 802.11 CSMA/CA medium access mechanism through the introduction of random short delays. *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, WiOpt 2015* **2015**, pp. 331–338. https://doi.org/10.1109/WIOPT.2015.7151090.
32. Lu, K.; Wang, J.; Wu, D.; Fang, Y. Performance of a burst-frame-based CSMA/CA protocol for high data rate ultra-wideband networks: Analysis and enhancement. *ACM International Conference Proceeding Series* **2006**, *191*. https://doi.org/10.1145/1185373.1185388.
33. Julier, S.; Uhlmann, J.; Durrant-Whyte, H. A new approach for filtering nonlinear systems. *Proceedings of 1995 American Control Conference - ACC'95* **1995**, *3*, 1628–1632 vol.3.
34. Merwe, R.; Wan, E. Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models. *Proceedings of the Workshop on Advances in Machine Learning* **2003**.
35. Schubert, R.; Richter, E.; Wanielik, G. Comparison and evaluation of advanced motion models for vehicle tracking. In Proceedings of the 2008 11th International Conference on Information Fusion, 2008, pp. 1–6.
36. Julier, S.; Uhlmann, J.K. A General Method for Approximating Nonlinear Transformations of Probability Distributions. Technical report, 1996.
37. Meyer, J.; Sendobry, A.; Kohlbrecher, S.; Klingauf, U.; Von Stryk, O. Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo. 2012, Vol. 7628, pp. 400–411. https://doi.org/10.1007/978-3-642-34327-8_36.
38. Bianchi, M.; Barfoot, T.D. UAV Localization Using Autoencoded Satellite Images. *IEEE Robotics and Automation Letters* **2021**, *6*, 1761–1768. https://doi.org/10.1109/LRA.2021.3060397.
39. Gupta, A.; Fernando, X. Simultaneous Localization and Mapping (SLAM) and Data Fusion in Unmanned Aerial Vehicles: Recent Advances and Challenges. *Drones 2022, Vol. 6, Page 85* **2022**, *6*, 85. https://doi.org/10.3390/DRONES6040085.
40. Keiller, J. Localization for teams of autonomous ground vehicles. M.s. thesis, University of Illinois at Urbana-Champaign, Champaign, IL, 2019.