IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 23 February 2024, accepted 5 March 2024, date of publication 18 March 2024, date of current version 22 March 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3377237

RESEARCH ARTICLE

WhereArtThou: A WiFi-RTT-Based Indoor Positioning System

REBAL JURDI[1], HAO CHEN[1], YUMING ZHU[1], (Member, IEEE), BOON LOONG NG[1], NEHA DAWAR[1], CHARLIE ZHANG[1], AND JOHNNY KYU-HUI HAN[2]

[1]Samsung Research America, Plano, TX 75023, USA
[2]Samsung Electronics, Gumi-si 39388, South Korea

Corresponding author: Rebal Jurdi (rebal.jurdi@samsung.com)

ABSTRACT Accurate indoor positioning is critical to a variety of use cases including tracking, proximity, mapping, and navigation. Existing positioning methods are either inaccurate (e.g. RSSI-based ranging), impractical (e.g. fingerprinting), or uncommon (e.g. Ultra-Wide Band/UWB). WiFi Round-Trip Time (RTT) marries the ubiquity of WiFi infrastructure with the accuracy of such ranging mechanisms as those used in UWB. We present *WhereArtThou*, a commercial-grade, plug-and-play indoor positioning solution (IPS) based on WiFi RTT. Our base algorithm uses an extended Kalman filter with a random walk motion model (EKF-RW) which relies solely on RTT distance measurements. We propose two EKF components to enhance positioning accuracy. First, as distance measurements can further be fused with inertial sensor readings, we propose a step-and-heading-based filter (EKF-SH) when such readings are available. We devise a method to fit the non-Gaussian step error with a Gaussian random variable to remain within the computationally-efficient Kalman filtering framework. Second, we define a distance-dependent measurement model to match the true statistics of the measurement noise and approach optimal position estimation. Moreover, we measure the gain from the proposed enhancements not only through the almost-exclusively-used error metric in the indoor positioning literature, the Euclidean distance and its variations, but also through metrics commonly used in satellite and maritime navigation, the cross-track and along-track errors, in addition to a set of metrics of our own definition. Finally, we show that RTT is highly susceptible to the human body holding the device measuring it, and we case-study the impact of human-body blockage on positioning and ranging errors. We test our algorithms on over 18 hours of walking data collected on different devices, in different locations, and with different users, and we observe that the EKF-RW and EKF-SH achieve 90th percentile distance errors of 1.65 m and 1.45 m, and 90th percentile cross-track errors of 0.85 m and 1.55 m, making our solution primed for commercial deployment.

INDEX TERMS Indoor positioning, indoor localization, wireless positioning, device-based positioning, range-based positioning, WiFi round-trip time (RTT), fine-timing measurement (FTM), ranging, Bayesian filter, Kalman filter, pedestrian dead reckoning (PDR), inertial measurement unit (IMU), ground truth.

I. INTRODUCTION

Indoor positioning has grown in popularity over the last decade in parallel with the growth in the number of personal wireless devices and in the size of wireless infrastructure [1], [2], [3]. While the use cases are plenty and include smart homes and buildings, surveillance, disaster management, industry and healthcare, they all require wide availability and good accuracy. Most of the existing positioning techniques suffer from one or more of the following drawbacks: inaccuracy, impracticality, and uncommonness. For example, fingerprinting is a two-step method that builds a spatial database mapping position descriptors to features such as received signal strength, e.g. RSSI in WiFi or Bluetooth, and then enables the online lookup of the position from the features [4], [5]. Building a database is cumbersome, and the

The associate editor coordinating the review of this manuscript and approving it for publication was Ángel F. García-Fernández.

slightest change to the floor layout may render the database useless and obsolete. Another example is positioning based on ultra-wide band (UWB), another wireless technology [6]. While UWB provides great accuracy, UWB *tags*, to be used as reference points, are far less common compared to WiFi, which can be found in almost every commercial and residential space. Enter WiFi *round-trip time (RTT)* which standardizes the *Fine Timing Measurement (FTM)* mechanism for accurate ranging inspired by UWB [7]. Given the pervasiveness of WiFi access points and devices, WiFi RTT has become the strongest contender to win the indoor positioning race.

An entirely orthogonal positioning philosophy is *dead reckoning*, where an object's position is estimated not through measured ranges with reference points, but rather through a continuous account of its displacement. *Pedestrian dead reckoning*, or *PDR*, refers specifically to when the object in question is a pedestrian walking indoors or outdoors [8]. With the proliferation of sensors inside smart devices, e.g. smartphones, tablets, and smartwatches, PDR has naturally evolved to supplement legacy wireless positioning technologies, such as WiFi, that have long been supported by these very devices [9], as well as more recent and less common technologies such as UWB. The inertial measurement unit (IMU) is an in-device hardware module that combines numerous sensors with functional differences, e.g. the accelerometer measures linear acceleration; the gyroscope measures angular velocity; the magnetometer measures the strength and direction of the magnetic field. These three sensors alone are able to estimate the object's velocity, i.e. speed and direction, and predict its trajectory by accruing its displacements through methods inspired by robot kinematics, and used in autonomous navigation and simultaneous localization and mapping (SLAM) [10]. While either range-based, wireless localization or sensor-based, PDR can be used exclusively, combining the two approaches through what is known as *sensor fusion* reduces uncertainty and establishes a good trade-off between the predicted trajectory and the observed one, as evidenced by its widespread use in indoor localization competitions [11], [12].

Recent years have seen an increase in work on range-based indoor positioning using WiFi RTT (FTM). Reference [13] investigates the quality of RTT distance measurements on different smartphones in different operational settings and under different link conditions. Reference [14] also studies the ranging quality of different devices in different configurations and characterizes the biases in ranging errors. This work also proposes a method to resolve the ambiguity in the estimated position due to the presence of multiple equally-plausible candidates. Reference [15] compares two-sided ranging, i.e. the FTM mechanism, with one-sided ranging using legacy APs that are not FTM-capable. Reference [16] implements an active monopulse radar system that uses the FTM mechanism through a single access point to measure distance. Reference [17] empirically builds a measurement likelihood function and uses particle filtering to estimate the position of a moving target. Reference [18] proposes RTT as a fingerprint instead of the traditionally-used RSSI. Reference [19] proposes calibration algorithms to overcome the impact of clock deviation and non-line-of-site (NLOS) links on ranging errors. Reference [20] also presents a calibration method to eliminate range offsets due to clock skew. Reference [21] studies ranging performance through the scope of different variables such as sampling and multipath. Reference [22] compares the ranging accuracy between WiFi RTT and UWB. Reference [23] proposes a positioning algorithm based on multi-dimensional scaling, by converting measurements of distances between wireless nodes, specifically RTT, into two-dimensional coordinates. Reference [24] proposes an RTT-based fingerprinting method based on Gaussian process regression, using particle swarm optimization to tune the model's hyperparameters. Reference [25] explains the enhancements as outlined in IEEE 802.11az to the existing ranging techniques and provides a detailed evaluation through experiments conducted in the mmWave (60 GHz) band. Last but not least, [26] proposes a method to distinguish LOS links from NLOS as well as error correction methods for each scenario.

Literature on PDR methods, and dead reckoning methods more generally, is as deep and as old as that on wireless methods, but literature that specifically combines PDR with RTT is rather limited. Recent work, e.g. [28], proposes positioning methods using sensor fusion and ranging error compensation models based on least squares fitting. The proposed positioning methods estimate the user's heading from RTT ranges, rotational velocity sensors, and the in-device magnet while combating distortions in the magnetic field. [29] uses a federated filter to fuse RTT measurements with sensor readings. Reference [30] proposes to use RTT measurements to estimate the user's rotation, achieving what the authors refer to as ''virtual inertial sensors''. Reference Recent work [31], [32] also combines RTT measurements with IMU sensor readings to estimate position through extended and unscented Kalman filters. One work [33] takes a geometric approach to combine RTT measurement models with PDR models into a system of equations. The authors then use not filtering methods, but rather optimization methods to solve for position. Most recently, [34] proposes a positioning method for smartphone-based robots using adaptive and error-state Kalman filters based on inertial navigation and sensor fusion.

Kalman filters, and more generally Bayesian filters, are the mainstay of positioning and tracking [10]. The Bayesian filter, and its efficient, linear counterpart, the Kalman filter, are probabilistic techniques that enable the estimation of the state of an observed dynamical system or the probability thereof. These filters assume that the trajectory of the system is expressed through a *motion model*, also known as a state transition model, which describes how the system evolves

across time. The measurement of the state is expressed through a *measurement model*, or an observation model, which relates the state, or its probability distribution, at a given time to measurements collected at that time. With an incoming stream of measurements, the state of the system is recursively estimated in two steps. In the first step, known as the *prediction step*, the current state is predicted from its most recent estimate solely using the motion model. In the second step, known as the *update step*, the predicted state is corrected with collected measurements. In this work, we highlight the shortcoming of measurement and motion models used in relevant work, and propose different ways to overcome these limitations.

To begin with, we propose an enhancement to measurement methods used in recent work, which are based on correcting measured RTT ranges, either by subtracting a back-off, error-like quantity, or by applying a transformation whose parameters are empirically fit. We introduce a model that not only corrects the range measurements in the average sense based on their value, but also weights them in proportion to their reliability: placing more weight on shorter ranges and less weight on longer ones. We refer the reader to Section IV-B for a thorough discussion.

Moreover, we consider two shortcomings of motion models: its states are unobservable, and uncertainties in its inputs are unaccounted for. We propose a model that drops unobservable states and properly accounts for input errors. We refer the reader to Section IV-C for a detailed discussion.

Furthermore, while the referenced papers use established techniques such as Bayesian filters combined with sophisticated PDR models, our target has been a simple and computationally-inexpensive solution that can run as smooth on flagship, computational power-horses as on lower-end, possibly embedded devices. These design drivers directly relegate three groups of algorithms, 1) iterative solvers, 2) non-linear Bayesian filters and filter banks, and 3) higher dimension models, in favor of a low-dimensional (linear) Kalman filter that achieves a speedup proportional to the number of iterations, particles, filters, or dimensions that would have otherwise been used.

We introduce *WhereArtThou*,[1] a device-side solution and key component of our indoor positioning system (IPS) based on WiFi RTT, that can achieve a 90th-percentile positioning error of under 1.6 m. Our solution can be readily deployed by the average user using commercial-off-the-shelf, FTM-capable smartphones and access points.

The contributions of this paper are as follows:

- A distance-dependent measurement model that maps RTT distance measurements to appropriate observation noise mean and variance, which are used in the Kalman filter update step; such a model matches the true statics of the observation noise and approaches optimal estimation
- A step error model that fits a non-Gaussian step error with a Gaussian random variable, upholding the computationally-efficient Kalman filter framework
- Detailed benchmark results of our algorithms through not only the Euclidean distance error metric or any of its derivatives, which is the the almost-exclusively used metric in indoor positioning literature, but also through commonly used metrics in satellite, aerial, maritime, and ground navigation
- A set of metrics to measure the jitter of position estimates along and across the track of motion
- A case study exposing the detrimental effect of the user's body on ranging and positioning accuracy

The algorithms described in this paper, including the different models introduced herein, can beat a 90-percentile positioning error of 2 m comfortably as evaluated through different metrics and tested on extensive datasets obtained with different devices and in different test sites.

The paper is organized as follows. In Section II we introduce and overview *Fine Timing Measurements (FTM)*, the WiFi mechanism that enables our solution. In Section III, we describe our data collection procedure. In Section IV, we detail our positioning algorithms and their novel aspects. In Section V, we describe the evaluation metrics. In Section VI, we benchmark our proposed algorithms. In Section VII we case study the impact of human body blockage on ranging and positioning accuracy. And finally, we conclude the paper in VIII.

II. FINE TIMING MEASUREMENT

While round-trip time (RTT) is an abstract, blanket term that measures the time it takes for an object to travel from source to destination and back again, the term RTT in this paper refers specifically to the *WiFi Fine Timing Measurement (FTM)* mechanism introduced through IEEE 802.11mc, standardized under IEEE 802.11-2016, and informally known as *WiFi RTT*.

Fine timing measurement (FTM), is a wireless network management procedure that allows a WiFi station (STA) to accurately measure the distance from another STA, e.g. an access point (AP), by measuring the round-trip time (RTT) of the frames exchanged between them.

An STA wanting to localize itself, known as the *FTM initiator (FTMI)*, with respect to other STAs, known as *FTM responders (FTMR)*, schedules an FTM session during which the STAs exchange messages and measurements. The FTM session consists of three phases: negotiation, measurement exchange, and termination.

In the negotiation phase, the FTMI negotiates with the FTMRs key parameters, such as frame format and bandwidth, number of bursts, burst duration, burst period, and number of measurements per burst. In the measurement phase, the FTMI measures the RTT of the frames exchanged with the FTMRs. The measurement phase consists of one or more bursts, and each burst is comprised of one or more (fine timing) measurements. In order to be useful for positioning

[1]The name *WhereArtThou* is a double play on words. One means "where are you" in archaic English. The other hints at *WiFi RTT* as they both start with the letter "W" followed by the letter string "RTT".

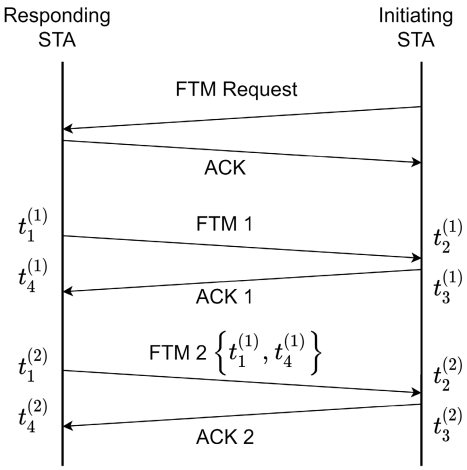


FIGURE 1. Example of an FTM measurement exchange with one burst and two measurements per burst.

and proximity apps, the RTT between two STAs is translated into a distance according to the following equation:

$$d = \frac{\text{RTT}}{2}c. \tag{1}$$

Each FTM of the burst will yield a distance sample, with multiple distance samples per burst. The samples can be averaged out, and their mean and standard deviation can be reported.

Given multiple FTM bursts and multiple measurements per burst, the distance samples can be combined in different ways to produce a representative distance measurement. For example, the mean distance and its standard deviation can be reported, or the median or some other percentile.

While FTM was conceived in the 11mc amendment, it underwent a series of enhancements through the 11az and 11bk amendments. IEEE 802.11az *Enhancements for Positioning*, dubbed *Next Generation Positioning*, targets enhancing scalability, security, accuracy, and efficiency by extending the maximum supported bandwidth to 160 MHz from 80 MHz, and adding support for the 6 GHz band and MIMO. The 11bk amendment further extends the maximum supported bandwidth to 320 MHz.

III. DATA COLLECTION

Establishing the ground truth of the user's varying position is critical to evaluating positioning algorithms and characterizing the errors between a sequence of positions and their estimates. We describe a simple and inexpensive method to define the ground truth trajectory of a moving user and to collect data.

Although vision-based motion capture systems like ViCon and OptiTrack can quite accurately capture the trajectory of motion [35], they are not critical for WiFi positioning as compared to other, more accurate technologies like UWB.. Positioning and ranging errors with WiFi RTT are on average far greater than the error between the vision-based trajectory and the timestamp-based one to be presented below. In UWB positioning, however, errors could be of the scale of the length of a smartphone, making an accurate ground truth indeed critical for evaluation. But even with UWB, the gap in RMSE (vision-based vs. timestamped) was observed to be in the 5-15 cm range at the time where the RMSE errors themselves are of that same order. With WiFi, however, the RMSE is in the meter range, so such an accuracy in the absolute error figure is not as critical. Moreover, experiments enabled by vision-based systems cannot be reproduced in subsequent work that cannot access similar systems, and results and findings cannot be validated.

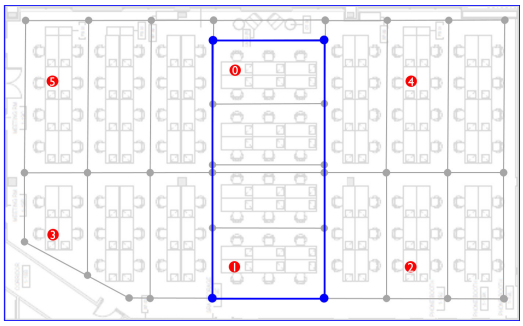


FIGURE 2. Traversable office space is converted to a grid network made of nodes and segments used to define walking routes along which the user's true position can be confidently computed. Gray lines and gray dots represent segments and nodes. The blue rectangle and blue dots on it represent a test route and turning points. Red dots represent APs.

We convert a floorplan into a grid network of *routes*, *segments*, and *nodes*. Every route is made of segments, and every segment is defined by two nodes used as anchors to determine the ground truth coordinates of the points along every segment. Nodes are chosen for the ease of inferring their Cartesian coordinates, e.g. the intersection of two corridors or aisles, and they are used as anchors to determine the ground truth coordinates of the points along every segment. A route can be closed if its extremities touch, and open, otherwise. Accordingly, closed routes are polygonal, and open routes are piece-wise linear. Every datapoint corresponds to the time series of data collected by the user's device as they traverse a designated route. The collected data includes RTT-related measurements and statistics, IMU sensor readings, and other useful information.

In the data collection phase, the user stands at the start point of a route and waits for a signal from the data collection app prompting them to start walking. When a user is about to make a turn, they press a "timestamp" button to mark the occasion. The user tries to maintain a uniform speed throughout each segment of the route. While the user is walking, the data collection app uses an Android API to make ranging requests[2] with registered FTM-capable access points, timestamps the corresponding ranging responses, and extracts the RTT distance measurements and relevant information.

[2]The ranging request is sent by the application layer down through the WiFi stack to start an FTM session the results of which are sent back up to the application layer as a ranging response.

In the evaluation phase, the underlying route of every datapoint is sampled into a discrete sequence of *waypoints* $\{\mathbf{x}_k\}$ at times $\{t_k\}$ at which ranging results are received. The true position of waypoints along a segment is computed through interpolating between the two nodes defining the segment whose true positions are known with close-to-perfect accuracy. For a waypoint occurring at an arbitrary time t and falling between any two consecutive nodes with positions $\mathbf{x}_1$ and $\mathbf{x}_2$ that the user visits at times t_1 and t_2, we compute the true position $\mathbf{x}$ according to the following equation:

$$\mathbf{x} = \mathbf{x}_1 + \left(\frac{t}{t_2 - t_1}\right) \cdot (\mathbf{x}_2 - \mathbf{x}_1). \tag{2}$$

IV. POSITIONING ALGORITHM

In this section, we describe our positioning algorithm and highlight two innovations therein: a distance-dependent measurement model and a step-and-heading motion model that fits into the Kalman filter framework.

A. RANDOM WALK EXTENDED KALMAN FILTER

We start by giving up any linear or angular kinematic information, e.g. acceleration and angular velocity, and rely solely on ranging (RTT) information to derive the position of the user/device. Later, we incorporate kinematic information provided by the device IMU to predict the user's trajectory and enhance positioning accuracy.

Trilateration, or *multi-lateration*, is the most basic method of computing the position of a device from *ranges*, i.e. distance measurements, with anchor points whose locations are known. This method solves a non-linear least square problem with the following objective:

$$f(\mathbf{x}) = \sum_{a=1}^{A} w_a \left(r_a - d(\mathbf{x}, \mathbf{p}_a)\right)^2, \tag{3}$$

where r_a is the range from anchor point, or access point (AP), a, $1 \leq a \leq A$, $\mathbf{p}_a$ is the location of AP a, $d(\cdot, \cdot)$ is the Euclidean distance function, and w_a are weights. Without loss of generality, we will assume throughout this paper that ranges with *all* the APs are available.

The problem above is typically solved using a minimization algorithm such as Levenberg-Marquardt, or relaxed, expanded, or reformulated into other problems and solved using methods such as semi-definite programming [36], the parallel projection method [37], or the projection on convex sets method [38], to name a few.

The methods above are limited in two ways. First, they are iterative, where every iteration involves numerous matrix operations, such as multiplication and inversion, which consume both time and power. Second, they are memoryless; the aforementioned methods estimate position from a set of ranges obtained during the same time step, forfeiting a great opportunity to use a vast past history of range measurements and position estimates. A Kalman filter, overcomes those two limitations.

A Kalman filter recursively estimates the state of a dynamical system from a sequence of measurements obtained over time and an underlying model of state evolution [10]. The filter models the underlying system by two linear equations, a motion, or state transition, equation, and a measurement, or observation, equation. In our positioning context, the motion model describes the evolution of position with time, and the measurement model describes the relationship between that position and the RTT ranges from the different APs.

Two common *sensor-free* motion equations in the indoor positioning literature are the *second-order* and *third-order* motion equations involving kinematic quantities such as velocity and acceleration.

In the second-order model, which we refer to as *position-velocity (PV)* model, the state of the system $\mathbf{s}_k$ is the two-dimensional device position and velocity, i.e. $\mathbf{s}_k = [\mathbf{x}_k^T, \mathbf{v}_k^T]$, and the state transition equation is

$$\begin{bmatrix}\mathbf{x}_k \\ \mathbf{v}_k\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \Delta t_k \mathbf{I} \\ \mathbf{O} & \mathbf{I}\end{bmatrix} \cdot \begin{bmatrix}\mathbf{x}_{k-1} \\ \mathbf{v}_{k-1}\end{bmatrix} + \varepsilon_k, \tag{4}$$

where ε_k represents the uncertainty in state variables, possibly capturing the impact of random acceleration on position and velocity, Δt_k is time between the last two estimation epochs, and $\mathbf{I}$ and $\mathbf{O}$ are the 2×2 identity and zero matrices.

In the third-order model, most recently used in a work on WiFi-RTT-based positioning [34], the state of the system $\mathbf{s}_k$ becomes the two-dimensional device position, velocity, and acceleration, i.e. $\mathbf{s}_k = [\mathbf{x}_k^T, \mathbf{v}_k^T, \mathbf{a}_k^T]$, and the state transition equation becomes

$$\begin{bmatrix}\mathbf{x}_k \\ \mathbf{v}_k \\ \mathbf{a}_k\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \Delta t_k \mathbf{I} & \frac{1}{2}\Delta t_k^2 \mathbf{I} \\ \mathbf{O} & \mathbf{I} & \Delta t_k \mathbf{I} \\ \mathbf{O} & \mathbf{O} & \mathbf{I}\end{bmatrix} \cdot \begin{bmatrix}\mathbf{x}_{k-1} \\ \mathbf{v}_{k-1} \\ \mathbf{a}_{k-1}\end{bmatrix} + \varepsilon_k, \tag{5}$$

where ε_k represents the uncertainty in state variables that are evolving with time.

In the absence of motion sensors in the device held by the user, the above models lead to unwanted undershooting and overshooting of the position estimate. Consider the two following scenarios.

In the first one, a user holds their device, walks along a linear path and abruptly stops once they reach the end. In the absence of an accelerometer to measure acceleration, the velocity as a state is slow to respond to sudden change in speed. During the ramp-up phase, the estimated position *undershoots* if the initial speed is set too small, i.e. it slowly catches up to the true position. When the user has been walking straight for some time, and if range measurements are accurate, the filter may be able to close-in on the true velocity, yet this does not overcome the next challenge. During the wind-down phase, the estimated position *overshoots* because the velocity as a state is non-zero, i.e. the position estimate goes past the true position.

In the second scenario, a user walks along a horizontal path and abruptly makes a sharp turn midway through.

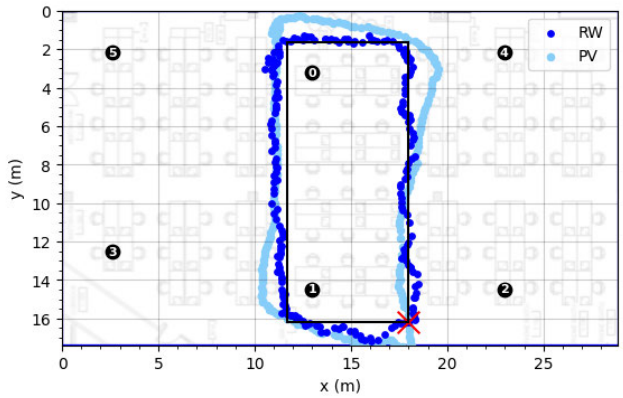


FIGURE 3. When the user makes a sharp turn, the position estimate overshoots past the corner when the position-velocity (PV) model is used.

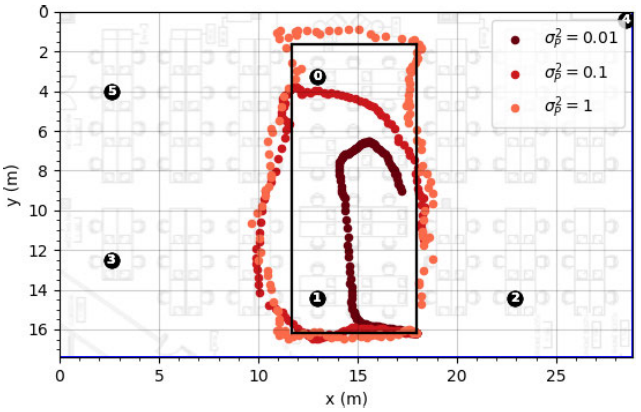


FIGURE 4. The process noise variance σ_P^2 can either restrict or liberate motion; a lower σ_P^2 gives a smooth trajectory, but results in round corners and lag; a higher σ_P^2, however, sharpens corners and catches up to the user, but gives a jumpy trajectory.

In the absence of an accelerometer to measure the device's acceleration, and a gyroscope to measure the device's rotation angle and infer the user's acceleration, the velocity as a state is slow to respond to sudden change in walking direction. As a result, the estimated position continues along the same horizontal path, only to get back on the right track once enough range measurements have been collected.

Unlike position which is a state observed by RTT range measurements, velocity is an unobserved state that also does not take any inputs, so it can be neither predicted nor corrected. Therefore, we opt for a motion model that avoids undershooting and overshooting by dropping velocity as a state variable and retaining position as the the only state. We term this model the *random walk (RW)* model. Fig. 3 depicts the two scenarios above, and compares the trajectory estimated by the position-velocity motion model and that estimated by the random walk model.

We begin by assuming that the user can take an arbitrary step, i.e. a step with a random size and direction, and we model the evolution of their position by a *drift-less Gaussian random walk* model in the 2D plane. Accordingly, the motion model becomes

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{v}_k, \tag{6}$$

where $\mathbf{x}_k$ denotes the current position, $\mathbf{x}_{k-1}$ denotes the previous position, and $\mathbf{v}_k$ denotes a (fractional) step of random size and direction, specifically $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$, where

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_P^2 \Delta t_k^2 & 0 \\ 0 & \sigma_P^2 \Delta t_k^2 \end{bmatrix}, \tag{7}$$

$\Delta t_k = t_k - t_{k-1}$ is the time between consecutive ranges, and σ_P^2 is the *process noise variance* parameter reflecting the user's average speed.

There are two key things to note about the random walk model. First, the longer the time between consecutive ranges, the farther away the user can stray from their last known position. Practically speaking, a low ranging rate spreads out position estimates, while a high ranging rate brings them together. Second, the smaller the process noise variance σ_P^2, the more constrained the motion. Figure 4 illustrates the physical significance of the model parameter σ_P^2. Practically speaking, a lower σ_P produces a smooth trajectory, but one that lags behind the user, more so if they are walking fast; a higher σ_P produces a choppy trajectory, but one that catches up to the user regardless of how fast they walk. Therefore, the value of σ_P can be chosen to either restrict or liberate the estimated trajectory. The choice of σ_P needs to balance the conflicting objectives of increasing both response time and smoothness. In this work, we assume a fixed value for σ_P that works reasonably well for a wide range of walking speeds. We defer to future work dynamically adjusting σ_P based on the device IMU.

The process noise variance needs to be commensurate with the use case and expected speed range. For example, a value of $\sigma_P^2 = 3$ corresponds roughly to a speed range of 3 mph, or 1 m/s, for walking, to 7 mph, or 3.3 m/s, for running. Alternatively, a value for $\sigma_P^2 = 2$ corresponds roughly to a lower speed range of 1 mph, or 0.4 m/s, for strolling, to 5 mph, or 2.2 m/s, for jogging. We explain how the process noise variance σ_P can be derived from the target speed range in the appendix.

As for the measurement model relating the latest set of ranges to the corresponding position, we assume the following additive white Gaussian noise model:

$$y_{a,k} = d(\mathbf{x}_k, \mathbf{p}_a) + w_{a,k}, \tag{8}$$

where $y_{a,k}$ is the kth range, i.e. RTT distance measurement, from AP $a = 1 \ldots A$, $d(\mathbf{x}, \mathbf{p}_a)$ is the ground truth distance between the device at position $\mathbf{x}_k$ and AP a at a location $\mathbf{p}_a$, and $w_{a,k}$ is the white Gaussian measurement noise that models the ranging error and is assumed to be uncorrelated across time and APs. Additionally, the measurement noise is assumed for the time being to have a fixed variance across time and APs, i.e. $w_{a,k} \sim \mathcal{N}(0, \sigma_M^2)$. We stack the ranges $\{y_{a,k}\}$ with the A APs to obtain the vector observation $\mathbf{y}_k$ at

time t_k

$$\mathbf{y}_k = \begin{bmatrix} y_{a,k} \\ \vdots \\ y_{A,k} \end{bmatrix} \tag{9}$$

with a corresponding measurement noise $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$, where

$$\mathbf{R}_k = \begin{bmatrix} \sigma_M^2 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sigma_M^2 \end{bmatrix}. \tag{10}$$

Having defined the motion and measurement models, we can now derive the Kalman filter *prediction* and *update* steps. In the prediction step, the Kalman filter predicts the next position $\hat{\mathbf{x}}_{k|k-1}$ from the last known positon estimate $\hat{\mathbf{x}}_{k-1}$ and computes its covariance $\mathbf{P}_{k|k-1}$ using the motion model (6). The prediction euqations are

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k-1}, \tag{11}$$

$$\mathbf{P}_{k|k-1} = \mathbf{P}_k + \mathbf{Q}_k. \tag{12}$$

The update step of a Kalman filter interpolates between the predicted position $\hat{\mathbf{x}}_{k|k-1}$ and the position as inferred from the observation $\mathbf{y}_k$. As the distance $d(\mathbf{x}_k, \mathbf{p}_a)$ is a non-linear function of the position $\mathbf{x}_k$, the relationship between the observation $\mathbf{y}_k$ and the position $\mathbf{x}_k$ needs to be linearized as follows to remain within the Kalman filter framework:

$$\mathbf{y}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{w}_k, \tag{13}$$

where the matrix $\mathbf{H}_k$, known as the observation matrix, has elements

$$[\mathbf{H}_k]_{a,j} = \frac{\partial d(\mathbf{x}_k, \mathbf{p}_a)}{\partial x_j}\Big|_{x_j = [\hat{\mathbf{x}}_{k|k-1}]_j}. \tag{14}$$

The linearization of the observation equation gives the *extended Kalman filter (EKF)* whose update equations are given by

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \left(\mathbf{y}_k - \mathbf{d}(\hat{\mathbf{x}}_{k|k-1}) \right), \tag{15}$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}, \tag{16}$$

where $\mathbf{d}(\cdot)$ is the vector of distances from the A APs, $\mathbf{I}$ is the 2×2 identity matrix, $\mathbf{K}_k$ is the *Kalman gain* and is a function of the measurement noise and *a priori* error covariance matrices $\mathbf{R}_k$ and $\mathbf{P}_{k|k-1}$, and $\mathbf{P}_k$ is the *a posteriori* error covariance matrix. The prediction and update equations are obtained following standard derivation [10].

B. DISTANCE-DEPENDENT MEASUREMENT MODEL

We propose a refinement of the measurement model to enhance positioning accuracy. The way a Kalman filter, or a Bayesian filter in general, works is that it first predicts the next position from the most recent estimated position according to the motion model, and then updates its prediction from the most recent measurement according to the measurement model. The uncertainty in the user's motion and error in the measured ranges are modeled by Gaussian random variables whose mean and variance are parameters that are themselves subject to error. Therefore, setting those parameters close to their true values is critical to the accuracy of the estimate. In this work, we focus on modeling the measurement noise mean and variance as quantities that vary with distance. Specifically, we draw an empirical relationship between the measurement noise mean and variance on one side and the RTT ranges on another using a wealth of collected data. We term this the *distance-dependent measurement model (DDMM)*.

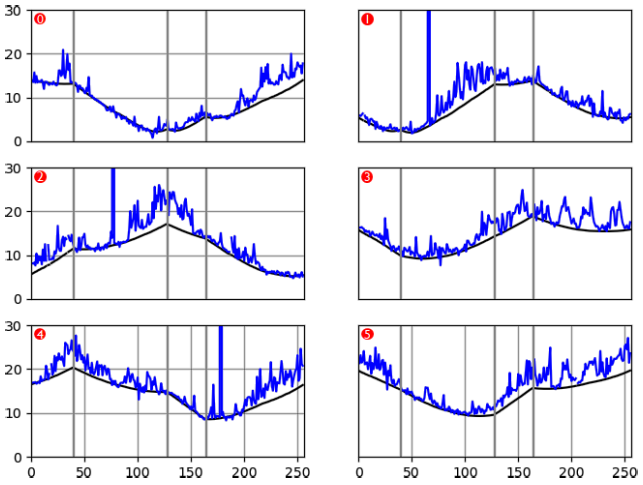


FIGURE 5. RTT distance measurements (blue) and ground truth distance (black) for different APs as a function of time corresponding to route `rc1` show in Fig. 2; vertical bars timestamp turning points along a route.

Fig. 5 shows an array of RTT distance measurements from 6 APs overlaid on top of the ground truth distance as a function of time. While the most obvious observation is that the ranges fluctuate and contain outliers, a more subtle observation is that these ranges are biased along certain parts of the route. The bias can be attributed to a consistent and dominant non-line-of-sight (NLOS) signal components due to reflectors in the environment which exaggerates the distance between the device and AP and results in a longer time of flight.

The measurement noise parameters, both mean and variance, can be characterized on an AP-by-AP basis. The resulting account, however, would remain specific not only to the site where the APs are deployed, but also to the AP locations within that site. Instead, we opt to make the measurement noise identically distributed across all APs.

Before putting the distance-dependent measurement model into operation (online phase), we train the model to give values to its parameters (offline phase). In the training phase, we obtain the RTT-distance-dependent measurement noise mean $\mu_M(y)$ and variance $\sigma_M^2(y)$ through the following four-step process:

1) Perform multiple rounds of data collection (see Section III) sampling all feasible locations
2) For every waypoint k of every route, compute the set of ranging errors $\{ w_{a,k} \}$ as the difference between the ranges $\{ y_{a,k} \}$ with the different APs $a = 1 \dots A$ and

the true distances $\{ d(\mathbf{x}_k, \mathbf{p}_a) \}$; plot the ranging errors $\{ w_{a,k} \}$ against $\{ y_{a,k} \}$ (see Figure 6)
3) Partition the range axis into bins of equal width, e.g. 1 m, and assign the ranging errors to their corresponding bins
4) Compute the mean and variance of the elements of every bin to obtain the sequences $\{ \mu_b \}$ and $\{ \sigma_b^2 \}$
5) Fit the sequence of means and variances, each with a straight-line, quadratic, or higher-order polynomial, to obtain the functions $\mu_M(\cdot)$ and $\sigma_M^2(\cdot)$ that give, for an arbitrary range y, estimates of the mean and variance of its corresponding ranging error

We observed that linear fitting is more suitable for fitting the mean of the ranging errors, and quadratic fitting is more suitable for fitting their variance as it increases more rapidly with distance than the mean.

Once the functions $\mu_M(\cdot)$ and $\sigma_M^2(\cdot)$ are defined, the distance-dependent model can now be used in the EKF update step. Under the new model, and for a received vector observation $\mathbf{y}_k = \left[y_{1,k} \cdots y_{A,k} \right]^T$, the measurement noise covariance matrix $\mathbf{R}_k$, originally defined in (10), is now computed as

$$\mathbf{R}_k = \begin{bmatrix} \sigma_M^2\left(y_{1,k}\right) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_M^2\left(y_{A,k}\right) \end{bmatrix}. \tag{17}$$

Moreover, the a posteriori estimate $\hat{\mathbf{x}}_k$, previously computed according to (15), is now computed as

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \cdot \mathbf{e}_k, \tag{18}$$

where the Kalman gain $\mathbf{K}_k$ is a function of the measurement noise covariance matrix $\mathbf{R}_k$, which itself is now a function of the measurement $\mathbf{y}_k$, and $\mathbf{e}_k$ is the *innovation*, i.e. the difference between the predicted range measurement and the actual one, is also a function of the measurement $\mathbf{y}_k$, namely

$$\mathbf{e}_k = \mathbf{y}_k - \begin{bmatrix} \mu_M\left(y_{1,k}\right) \\ \vdots \\ \mu_M\left(y_{A,k}\right) \end{bmatrix} - \mathbf{d}\left(\hat{\mathbf{x}}_{k|k-1}\right) \tag{19}$$

The distance-dependent EKF update essentially does two things. First, it compresses the range measurement $y_{a,k}$ by subtracting from it the estimated bias $\mu_M(y_{a,k})$ of the corresponding ranging error. Second, it assigns a weight, or confidence, appropriate to the value of the range. Bigger ranges are assumed more erroneous and are assigned smaller weights, while smaller ranges are assumed more truthful and are assigned larger weights. This is in agreement with the fact that NLOS links are more probable at greater distances.

Recent investigation of the ranging accuracy of WiFi RTT [15], [39] also observed that the spread of ranging errors and their average increases with distance, but they modeled these relationships through probability distributions and used those in the update step of a more general, Bayesian estimation framework, which is costly. Recent investigation of the positioning accuracy of WiFi RTT [27] proposed a correction of RTT range measurements by subtracting a back-off term to compensate for the typically-positive ranging error. The authors defined the back-off as a polynomial function of the RTT ranges whose coefficients are determined offline by fitting them to training data. A similar and more recent investigation proposed a better-educated error compensation method [26]. The method first identifies if the link is LOS or NLOS using RSSI measurements, and then applies the appropriate back-off term that is also a polynomial function of the RTT ranges. Another recent work on the positioning accuracy of WiFi RTT proposed a correction of RTT ranges by applying to them a polynomial function whose coefficients are determined through fitting [28].

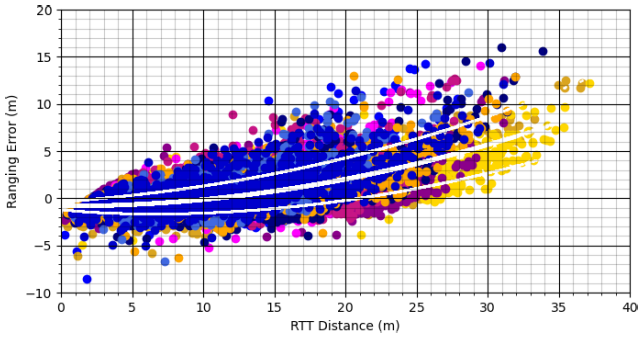


FIGURE 6. The bigger the distance, the worse the measurement; ranging errors are fitted with a quadratic function of the RTT distance from a single AP; dots correspond to waypoints along a route; colors indicate test routes. The white curve at the center describes the fitted mean μ_M of the ranging errors; the two white curves on either side are the mean ± the standard deviation σ_M.

Applying a corrective offset to RTT measurements does only half of the job. In a Kalman filter, or Bayesian filter in general, there are two key measurement-related quantities that go into the correction step, also known as update step: the noise covariance matrix $\mathbf{R}_k$ and the innovation $\mathbf{e}_k$. Processing the RTT measurement $\mathbf{y}_k$ by either applying it to a function, or by applying to it an additive corrective offset, may only bias the innovation $\mathbf{e}_k$ around zero. It does not, however, weight the different measurements according to their reliability. Those weights, in fact, are calculated from the measurement noise covariance matrix $\mathbf{R}_k$ and applied through the Kalman gain $\mathbf{K}_k$, which decides which of the following two contributes more to a reliable position estimate: the prediction or the observation.

Therefore, the measurement model that we introduce not only corrects the RTT measurements in the average sense based on their value, but also weights them appropriately: placing more weight on shorter distances and less weight on longer ones.

Finally, it is worth noting that the statistics of the measurement noise vector, its mean and covariance, are time varying. As the user moves around, their distance to every AP varies. As they move closer to an AP, the mean and variance of the measurement noise decrease. As they move away from an AP, the mean and variance increase. This behavior

is captured by the empirically-motivated distance-dependent model. In reality, the measurement noise vector is not only time varying, but also correlated across time and across links, making the measurement noise both colored and one with a non-diagonal covariance matrix. Properly characterizing the joint probability distribution of the measurement noise across time and across links can significantly improve positioning accuracy.

C. STEP AND HEADING EXTENDED KALMAN FILTER

Positioning accuracy can be enhanced by using sensor readings to account for incremental changes in position.

Dead reckoning is a method of estimating the position of a moving object by adding incremental displacements to its last known position. *Pedestrian dead reckoning*, or *PDR*, refers specifically to the scenario where the object in question is a pedestrian walking in an indoor or outdoor space. With the proliferation of sensors inside smart devices, e.g. smartphones, tablets, and smartwatches, PDR has naturally evolved to supplement wireless positioning technologies that have been long supported by these devices such as WiFi and cellular service, as well as more recent and less common technologies such as ultra-wide band (UWB). The inertial measurement unit (IMU) is a device that combines numerous sensors with functional differences, e.g. the accelerometer measures linear acceleration; the gyroscope measures angular velocity; the magnetometer measures the strength and direction of the magnetic field. Combining IMU sensor data and ranging measurements from wireless chipsets like WiFi and UWB, known as sensor fusion, can improve positioning accuracy by reducing uncertainty.

Methods that use PDR can be broadly split into two categories: inertial navigation (IN) methods, and step and heading (SH) methods [40].

IN methods continuously track the position of the device and its orientation, i.e. the direction it is facing in two- or three-dimensional space (also known as attitude or bearing). To determine the instantaneous position of the device, IN methods integrate the 3D acceleration to obtain velocity, and then the velocity to determine the displacement from the start point. To determine the instantaneous orientation of the device, IN methods integrate the angular velocity from the gyroscope to obtain the change in angles from the initial orientation. Measurement noise and biases at the levels of the accelerometer and gyroscope leads to a linear growth of orientation offset across time due to the integration of rotational velocity, and to quadratic growth of displacement error across time due to the double integration of the acceleration. Overcoming measurement noise and biases often demands a computationally-complex filter with a high-dimensional state vector that tracks all the biases and noise
statistics [41].

Unlike IN methods that track the position of the device continuously, SH methods update the device position less frequently by accumulating the steps that user takes from a start point. Every step is described as a vector whose magnitude is the size of the step and whose argument is the heading of the step. Instead of directly integrating sensor readings to compute displacement and change in orientation, SH methods perform a sequence of operations. First, an SH system detects a step or stride from the device's three-dimensional acceleration using one of many different algorithms, e.g. peak detection, zero-crossing detection, or template matching. Second, once a step or stride is detected, the SH system estimates the size, or length, of the step from the sequence of acceleration readings coinciding with the step. Third, the SH system estimates the heading of the step using the gyroscope, magnetometer, or a combination of both.

While step detection is prone to mis-detection, for e.g. due to peaks in acceleration falling below a detection threshold, as well as false alarm, for e.g. due to double peaks, SH methods remain more robust than IN methods in the absence of an accurate estimate of the statistics of the noise at the level of the different sensors. Therefore, we opt for an SH-based motion model.

Motion models based on sensor fusion, including step and heading models, are more rife in the indoor positioning and robotics literature than sensor-free motion models. In one common model, the state of the system $\mathbf{s}_k$ is the motion heading direction and two-dimensional position, i.e. $\mathbf{s}_k = [\mathbf{x}_k^T, \theta_k]$, and the motion model is

$$\begin{bmatrix} \mathbf{x}_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{k-1} + s_k \cdot \mathbf{g}(\theta_{k-1}) \\ \theta_{k-1} + \phi_k \end{bmatrix} + \mathbf{v}_k, \tag{20}$$

where

$$\mathbf{g}(\theta_k) = \begin{bmatrix} \cos\theta_k \\ \sin\theta_k \end{bmatrix}, \tag{21}$$

and $\mathbf{v}_k$ is the *additive* process uncertainty vector, s_k is the size of the detected step, and ϕ_k is the *differential* step heading, i.e. rotation angle of the device held by the user around the vertical axis in the global frame of reference. The step size s_k is computed from the three-dimensional acceleration through analytical models such as those defined in [42], [43], and [44], and the rotation angle ϕ_k from the three-dimensional angular velocity or from the change in magnet-based device orientation. Closed-form models, however, need not be used at all. Instead, a regression model trained on the time series of IMU sensor readings through supervised learning can be used [45], [46], [47], [48], [49]. The step size s_k and differential step heading ϕ_k are the system *inputs*, i.e. they act on the state $\mathbf{s}_k$ comprised of the position $\mathbf{x}_k$ and motion heading θ_k, and they change it. Those inputs, however, are prone to error.

The model above has been very recently borrowed and applied to WiFi-RTT-based positioning, e.g. in [27] and [28]. The model was amended by expanding the state to include a multiplicative step size correction factor α_k, leading to the

four-dimensional state $\mathbf{s}_k = [\mathbf{x}_k^T, \theta_k, \alpha_k]$ and motion model

$$\begin{bmatrix} \mathbf{x}_k \\ \theta_k \\ \alpha_k \end{bmatrix} = \begin{bmatrix} \mathbf{x}_{k-1} + \alpha_{k-1} \cdot s_k \cdot \mathbf{g}(\theta_{k-1} + \phi_k) \\ \theta_{k-1} \\ \alpha_{k-1} \end{bmatrix} + \mathbf{v}_k. \quad (22)$$

Existing models including (20) and (22) have two shortcomings: the states lack observability, and the state transitions lack specificity. Some state variables, e.g. the heading θ_k, are unobserved, and errors in system *inputs*, namely the step size s_k and rotation angle ϕ_k, are unaccounted for.

Motion models, e.g. (20) and (22), are non-linear, so they need to be linerized to remain within the efficient Kalman filtering framework. Generic linearization techniques such as the extended Kalman filter (EKF) and the unscented Kalman filter (UKF) produce new transition equations. The step that the user takes, consisting of the step size s_k and differential step heading ϕ_k, is the input to the system that acts on and changes its state, comprised of the user's position and their heading. Those inputs, however, are prone to error. Generic linearization recipes translate the non-linear relationships between states and inputs into linear ones, but the errors in the different inputs are not explicitly accounted for. Instead, they are arbitrarily swept under the umbrella of the state error term $\mathbf{v}_k$ that is linearly added to the state variable $\mathbf{s}_k$.

To overcome the limitations described above, we propose a state transition and error model that drops unobserved states and properly accounts for input errors. In Section VI, we compare our model with the two models (20) and (22) by plugging them into Kalman filters and evaluating their positioning accuracy.

Suppose that we can infer the the size of the step (or average step size) s_k taken between t_{k-1} and t_k as well as the step heading θ_k taken during that same time. Instead of the random walk EKF whose motion model takes no inputs, an enhanced EKF with a motion model that takes the step size and heading s_k and θ_k as inputs can be used to predict the next position. However, errors in sensor readings will propagate into the step size and heading and need to be modeled accordingly. A simple model accounts for additive noise to both the step size s_k and step heading θ_k as follows:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \left(s_k + v_{S,k}\right) \cdot \begin{bmatrix} \cos\left(\theta_k + v_{\theta,k}\right) \\ \sin\left(\theta_k + v_{\theta,k}\right) \end{bmatrix}, \quad (23)$$

where $v_{S,k} \sim \mathcal{N}(0, \sigma_S^2)$ is the additive Gaussian error in step size, and $v_{\theta,k} \sim \mathcal{N}(0, \sigma_\theta^2)$ is the additive Gaussian error in step heading. Eq. (23) is non-linear in the error variables $v_{S,k}$ and $v_{\theta,k}$, so the effective process uncertainty, which characterizes the error in the state, will not be Gaussian. Without a Gaussian process uncertainty, the computationally-efficient Kalman filtering framework will no longer apply. To overcome this, we are faced with two options. The first option is to use the more general Bayesian framework which supports non-linear equations and non-Gaussian errors. Under this framework, the state transition equation becomes a state transition probability, and the observation equation becomes an observation likelihood function [10]. An example of such a framework is known as *Monte-Carlo localization*, which uses a particle filter to approximate the probability distribution of the estimate by a set of particles, each having a value denoting position and a weight denoting probability. Running this framework, however, is computationally expensive, and it risks draining the battery and throttling other running applications. This brings us to the second option which is to transform the non-Gaussian process uncertainty into a Gaussian random variable which would then make the Kalman filtering framework applicable once again. In this case, Eq. (23) above would need to be transformed into the following:

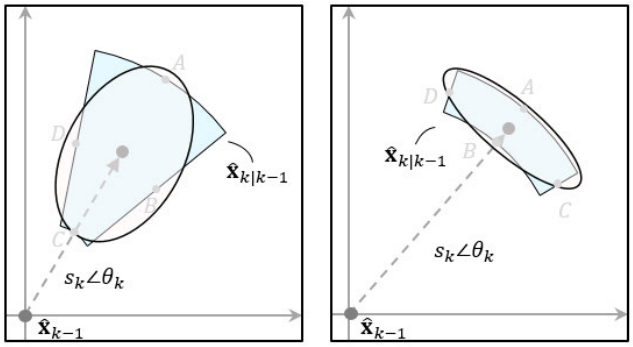


FIGURE 7. "Normalizing" the step error: Fitting the support of the step error with an ellipse; $\hat{\mathbf{x}}_{k-1}$ is the latest estimate; the prediction $\hat{\mathbf{x}}_{k|k-1}$ falls inside an annular sector; left: "tall" error with small σ_θ and large σ_S; right: "fat" error with large σ_θ and small σ_S.

$$\mathbf{x}_k = \mathbf{x}_{k-1} + s_k \cdot \begin{bmatrix} \cos\theta_k \\ \sin\theta_k \end{bmatrix} + \mathbf{v}_k, \quad (24)$$

where $\mathbf{v}_k$ is a Gaussian random variable representing the error in the step and accounts for both errors in the step size and direction. We will refer to Eq. (24) as the *step and heading (SH)* equation. Our remaining task is to "normalize" the non-Gaussian step error, i.e. to derive a transformation of the non-Gaussian step error into a Gaussian. We note that the step error according to Eq. (23) is concentrated inside an *annulus sector*, which is a cut of an annulus. We propose to capture most of the density of the error distribution by fitting the annulus sector with an ellipse, rotated and dilated appropriately. We propose the following algorithm.

We define the points A, B, C, and D around the most recent estimate $\hat{\mathbf{x}} = (\hat{x}, \hat{y})$ as follows:

$$A = \left(\hat{x} + (s + \sigma_S) \cdot \cos\theta, \quad \hat{y} + (s + \sigma_S) \cdot \sin\theta\right), \quad (25)$$
$$C = \left(\hat{x} + (s - \sigma_S) \cdot \cos\theta, \quad \hat{y} + (s - \sigma_S) \cdot \sin\theta\right), \quad (26)$$
$$D = \left(\hat{x} + s \cdot \cos(\theta + \sigma_\theta), \quad \hat{y} + s \cdot \sin(\theta + \sigma_\theta)\right), \quad (27)$$
$$B = \left(\hat{x} + s \cdot \cos(\theta - \sigma_\theta), \quad \hat{y} + s \cdot \sin(\theta - \sigma_\theta)\right). \quad (28)$$

We now use the four points A, B, C, and D to describe the two orthogonal axes of an ellipse, and then compute the variance of the horizontal and vertical errors σ_X^2 and σ_Y^2, as well as the their auto-covariance σ_{XY}. We compute σ_X^2, σ_Y^2,

and σ_{XY} as follows:

$$\sigma_X^2 = \left(\frac{\overline{DB}}{2}\right)^2 \cdot \sin^2\theta + \left(\frac{\overline{AC}}{2}\right)^2 \cdot \cos^2\theta, \tag{29}$$

$$\sigma_Y^2 = \left(\frac{\overline{DB}}{2}\right)^2 \cdot \cos^2\theta + \left(\frac{\overline{AC}}{2}\right)^2 \cdot \sin^2\theta, \tag{30}$$

$$\sigma_{XY} = \left(\left(\frac{\overline{DB}}{2}\right)^2 - \left(\frac{\overline{AC}}{2}\right)^2\right) \cdot \sin\theta\cos\theta, \tag{31}$$

where $\overline{AC}$ and $\overline{BD}$ are the lengths of the segments AC and BD. Finally, we can now build the covariance matrix $\mathbf{Q}_k$ of $\mathbf{v}_k$ as follows:

$$\mathbf{Q}_k = \begin{bmatrix} \sigma_X^2 & \sigma_{XY} \\ \sigma_{XY} & \sigma_Y^2 \end{bmatrix} \tag{32}$$

The step and heading model gives rise to a different filter, the *step and heading extended Kalman filter*, or *EKF-SH* for short. The EKF-SH has the same update step as the EKF-RW, as they both use the same observation model, but it has a different prediction step. Indeed, the prediction step of the EKF-SH is given by the following two equations:

$$\hat{\mathbf{x}}_{k|k-1} = \hat{\mathbf{x}}_{k-1} + s_k \cdot \begin{bmatrix} \cos\theta_k \\ \sin\theta_k \end{bmatrix}, \tag{33}$$

$$\mathbf{P}_{k|k-1} = \mathbf{P}_k + \mathbf{Q}_k. \tag{34}$$

In Section VI-B, we compare the EKF-SH with the EKF-RW through various error metrics.

V. EVALUATION METRICS

Numerical evaluation of an algorithm's positioning accuracy does not always stand up to visual evaluation which is what truly represents what the user experiences and sees on the screen. Often times, the EKF-SH trajectory appears better to the eye than the EKF-RW trajectory but scores less on the positioning error metric defined as the Euclidean distance. One possible reason is that the EKF-SH trajectory is prone to the "group delay" effect where the position estimates have a constant offset from the ground truth throughout the entire route. Most anticipated use cases of indoor positioning are not time-sensitive, so a position estimate that stays on track is arguably more important than one that catches up to the user's true position as they move but jumps around a lot. The disconnect between the user-perceived experience and the Euclidean distance as a standard error metric motivates the use of a more fair metric that measures the distance between the position estimate and the route rather than with the true position along that route. Fortunately, such metrics are common in satellite, aerial, maritime, and ground navigation. Hence, we will evaluate our different algorithms through the following metrics:

- **Horizontal error (HE):** Distance (Euclidean) between the true position $\mathbf{x}_k$ and its corresponding estimate $\hat{\mathbf{x}}_k$, i.e.

$$\mathsf{HE}_k = \left\| \hat{\mathbf{x}}_k - \mathbf{x}_k \right\|. \tag{35}$$

- **Along-track error (ATE):** Distance between the true position $\mathbf{x}_k$ and the orthogonal projection of its estimate $\hat{\mathbf{x}}_k$ on the trajectory, mathematically defined as

$$\mathsf{ATE}_k = \left\| \text{proj}_{(\mathbf{x}_{k-1},\mathbf{x}_k)}\hat{\mathbf{x}}_k - \mathbf{x}_k \right\|. \tag{36}$$

- **Cross-track error (XTE):** Distance between the position estimate $\hat{\mathbf{x}}_k$ and the true trajectory, defined as

$$\mathsf{XTE}_k = \left\| \hat{\mathbf{x}}_k - \text{proj}_{(\mathbf{x}_{k-1},\mathbf{x}_k)}\hat{\mathbf{x}}_k \right\|. \tag{37}$$

In addition to the standard error metrics defined above, we introduce 3 additional metrics that reflect user experience. These metrics are derived from standard error metrics, namely the XTE and ATE.

- **Swaying range (SR):** Twice the standard deviation of cross-track errors $\{\mathsf{XTE}_k\}$ of waypoints along a route. It captures the transverse jitter, i.e. the amount of variation across the track of motion. While the mean XTE can be zero, meaning that the estimate of the user's position is, on average, neither to the left nor to the right of the true position, the estimates can very much fluctuate inwards and outwards, similar to a car swaying side to side due to suspension faults.
- **Rocking range (RR):** Twice the standard deviation of the along-track errors $\{\mathsf{ATE}_k\}$ of waypoints along a route. It captures the longitudinal jitter, i.e. the amount of variation along the track of motion. While the mean ATE can be zero, meaning that the estimate of the user's position is, on average, neither leading nor trailing the true position, the estimates can very much fluctuate back and forth, similar to a car rocking back and forth when stalling as it comes to a halt.
- **Positioning lag (PL)**: It captures how far in time the position estimate is lagging behind the true position (positive value), or leading ahead of it (negative value). We define the positioning lag at waypoint k as:

$$\mathsf{PL}_k = \frac{\mathsf{ATE}_k}{u_k}, \tag{38}$$

where u_k is the speed at waypoint k.

To compute the swaying range, rocking range, and positioning lag, the XTE and ATE need to be redefined to give the estimated trajectory an aspect of *sidedness*, i.e. whether the position estimate falls on the left side of the true position or on the right side, and whether the estimate falls on the front side of the true position or on the back side. The XTE and ATE thus become *signed* quantities according to the following new definitions:

- ATE: Has a positive sign if the position estimate is trailing behind the true estimate and a negative sign if it is leading ahead, namely

$$\mathsf{ATE}_k = -\left\| \hat{\mathbf{p}}_k - \mathbf{x}_k \right\| \cdot \text{sgn}\left\langle \hat{\mathbf{x}}_k - \mathbf{x}_k, \mathbf{x}_k - \mathbf{x}_{k-1} \right\rangle \tag{39}$$

where $\hat{\mathbf{p}}_k = \text{proj}_{(\mathbf{x}_{k-1},\mathbf{x}_k)}\hat{\mathbf{x}}_k$, sgn $(\cdot)$ is the sign function, and $\langle\cdot,\cdot\rangle$ is the dot product.

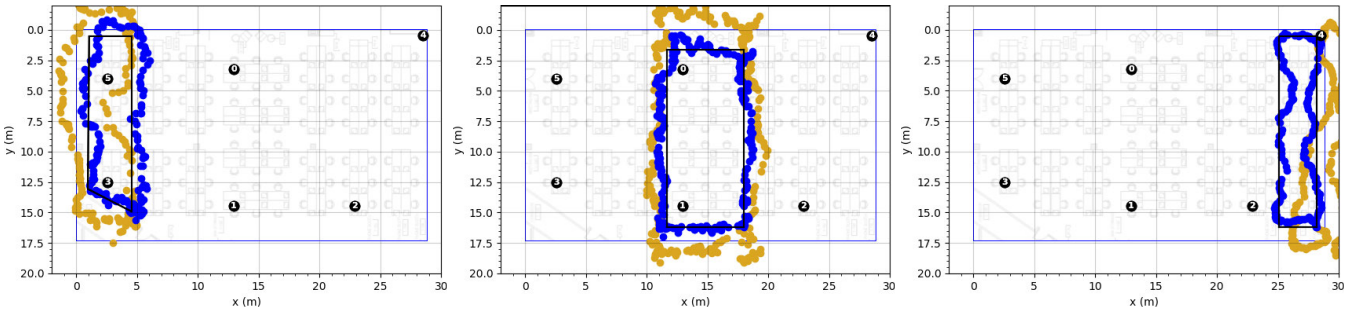
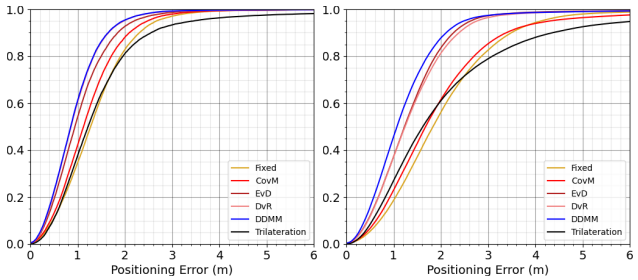


FIGURE 8. The distance-dependent measurement model (blue) pulls the estimated trajectory closer to the ground truth trajectory (solid black line) than the estimate under fixed measurement noise variance.



(a) Model A (b) Model B

FIGURE 9. The distance-dependent model adds a significant performance gain. This enhancement reduces the positioning error by up to 80cm on Model A and 1.4m on Model B.

- XTE: Has a positive sign if the position estimate is on the right side of the true trajectory and a negative sign if it is on the left side, namely

$$\mathrm{XTE}_k = -\left\| \hat{\mathbf{x}}_k - \hat{\mathbf{p}}_k \right\| \cdot \mathrm{sgn}\ \det\ \left[\mathbf{x}_k - \mathbf{x}_{k-1} \quad \hat{\mathbf{x}}_k - \mathbf{x}_{k-1}\right], \tag{40}$$

where $\det(\cdot)$ is the matrix determinant.

VI. EVALUATION

To evaluate our algorithms, we collected extensive data at 3 test sites. Every datapoint consists of a time series of RTT ranging responses and IMU sensor readings logged continuously while the test subject walks along a test route. More than 15 people participated in the data collection efforts, and more than 100 distinct routes were defined. In total, we produced more than 80 datasets consisting of more than 1,200 datapoints and north of 340,000 waypoints worth more than 18 hours of non-stop walking. We used 4 different devices, the Samsung Galaxy S20, S21 Ultra, S23, and XCover6 Pro. We also used 3 different AP models, the Google WiFi, Nest WiFi, and Aruba A515, all supporting FTM in the 5 GHz band over a bandwidth of 80 MHz.

The bulk of the data was collected in settings rife with signal blockages and reflectors: two offices and a tightly-packed exhibition room. The offices contained floor-to-ceiling columns and elevated height-adjustable desks acting as objects of signal blockage. The desks were mounted with desktop computers, laptops, and computer screens acting as signal reflectors.

In short, the collected data is sufficiently diverse, sampling numerous variables including:

- **Walking pattern**: Also known as *gait*, this includes parameters such as walking speed, step length and duration, stride length and duration. 15 test subjects participated in the data collection campaign.
- **FTM initiator**: The mobile device held by the user whose location needs to be identified. 4 models of mobile devices supporting the FTM technology have been used to collect data, for a total of more than 10 devices. FTM is a new technology that has not yet become a standard feature in mobile phones, and the market of FTM-capable mobile devices is currently limited to a handful of vendors at most.
- **FTM responder**: The WiFi access points with known positions acting as ranging partners for the mobile devices. 3 models of access points supporting the FTM technology have been used to collect data. The support of FTM by WiFi access points remains rather limited but is steadily rising, mirroring the trend of FTM support by mobile devices.
- **Traffic**: This includes both foot traffic and wireless network traffic, which not only are correlated, but also have correlated effects on positioning accuracy. More people create more line-of-sight blockages, which leads to the overestimation of round-trip time and distance. More people also operate more wireless devices, which reduces channel access for FTM initiators and responders, increases their chance of collision, and causes interference. All of the above lessens successful FTM exchanges and reduces measurement rate.

We evaluate the different algorithms and their variations on data collected on two representative smartphone models, Model A and Model B.

A. DISTANCE-DEPENDENT MEASUREMENT MODEL

We first evaluate the distance-dependent measurement model (DDMM) by benchmarking it against five baselines. The first baseline is trilateration, which, for every vector observation $\mathbf{y}_k$, initializes the solution as the centroid of AP positions. The second baseline is an EKF-RW which uses a fixed measurement noise variance $\sigma_M^2 = 3$ throughout the experiment.

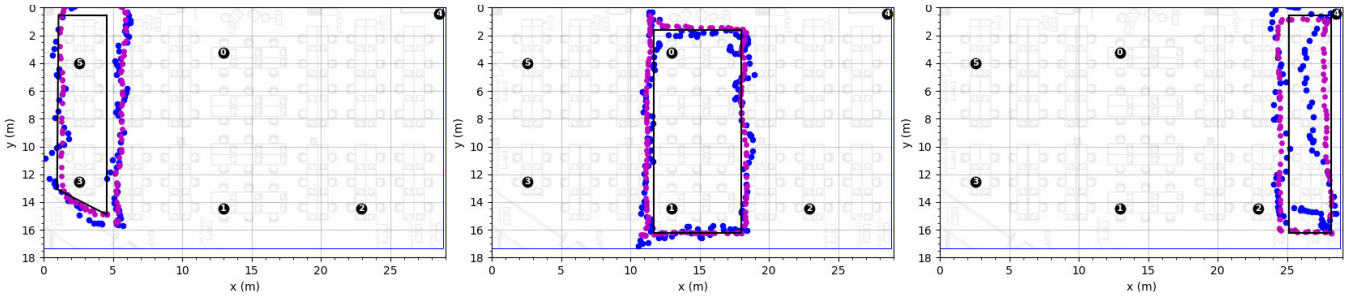
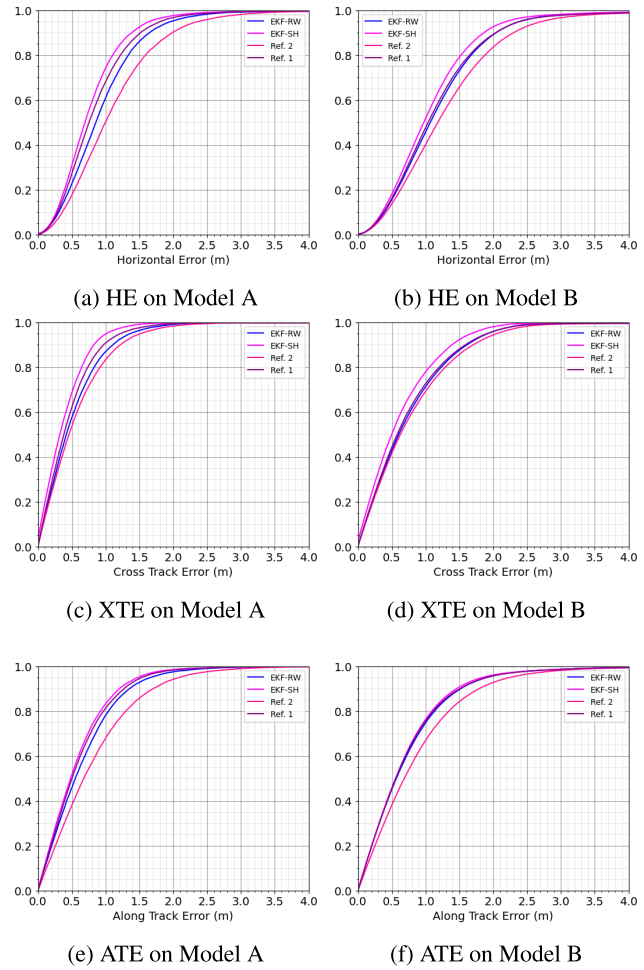


FIGURE 10. The trajectory estimated under the EKF-SH (purple) is closer to the ground truth trajectory (solid black line) than that produced under the EKF-RW (blue). The EKF-SH trajectory also shows discrete, almost equidistant dots corresponding to regular steps.



(a) HE on Model A (b) HE on Model B

(c) XTE on Model A (d) XTE on Model B

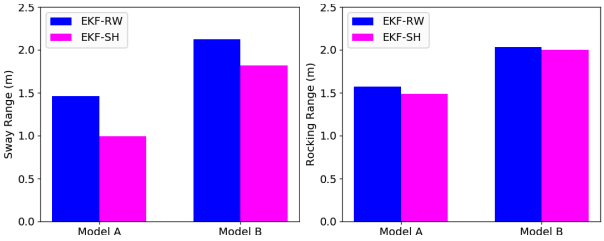
(e) ATE on Model A (f) ATE on Model B

FIGURE 11. The SH reduces the different error metrics by different amounts on different devices. The HE, i.e. the Euclidean distance, decreases under the EKF-SH by up to 30 cm in the 90th percentile, the XTE by up to 40 cm, and the ATE by up to 20 cm.

The third baseline, known as *covariance matching*, estimates the measurement noise variance for every AP a from the measurement post-fit residuals defined in the standard way as [50]

$$e_{a,k} = y_{a,k} - d(\hat{\mathbf{x}}_k, \mathbf{p}_a). \tag{41}$$

The fourth baseline, proposed in [27] (tagged "EvD"), corrects every RTT range measurement y_k by subtracting



(a) Swaying Range (b) Rocking Range

FIGURE 12. The SH motion model reduces the sway range by up to 60 cm and the rocking range by up to 20 cm.

an error term e_k that is a polynomial function of y_k. The coefficients of this polynomial are obtained by least-squares fitting of collected RTT measurements with known ground-truth values. The fifth baseline, proposed in [28] (tagged "DvR"), process every RTT range measurement y_k to estimate the true distance d_k by evaluating a polynomial function of y_k. The coefficients of this polynomial are also obtained by fitting collected measurements with know ground-truths.

We first compare the four different configurations *numerically* via the cumulative distribution function (CDF) of the positioning error, defined as the Euclidean distance between the true position of the device and its estimate. We make the following observations from Fig. 9. First, EKF-RW with a fixed measurement noise variance (base EKF-RW) outperforms trilateration only at higher error percentiles for some device models. Second, the covariance matching method achieves a moderate gain over the base EKF-RW on both models. Third, the EKF-RW armed with the distance-dependent measurement model outperforms covariance matching across the entire curve, especially at higher percentiles of the positioning error, where it achieves a gain of up to 1.5 m in the 90th percentile on certain device models. Furthermore, the DDMM outperforms EvD and DvR across the entire positioning error curve and for both smartphone models. While DDMM achieves a modest gain in positioning accuracy of 20 cm at any percentile on one smartphone model, it achieves a gain over EvD and DvR of up to 20 cm at just the 50th percentile (median), and up to 40 cm in the 90th percentile.

We then compare *visually* the positioning accuracy of two EKF-RWs: one that uses DDMM, and the base EKF-RW

that uses fixed measurement noise variances throughout. We overlay the scatterplot of the trajectory estimated by DDMM (in blue) on top of that estimated by the base EKF-RW (in yellow). We make the following observation. The base EKF-RW, which takes the measured ranges at face value, exaggerates the size of the trajectory as can be seen in the center box of Figure 8 in some situations, and pushes the trajectory outwards in some other cases. Meanwhile, under the DD model, the EKF update step first downsizes the measured RTT distance $y_{a,k}$ by removing an offset $\mu_M(y_{a,k})$, and then gives the observation a weight that is inversely related to the variance $\sigma^2_M(y_{a,k})$.

Looking at the device model as a test variable, we observed that the positioning accuracy on different models can be quite different. In one experiment, we evaluated the accuracy of the Random Walk EKF on two different phone models, Model A and Model B, over multiple datasets, and we observed that, under the distance-dependent model, the positioning error on Model A is less than that on Model B by 20 cm in the 50th percentile (median) and 40 cm in the 90th percentile. The performance gap widens significantly to 50 cm in the 50th percentile and 140 cm in the 90th percentile when a less-informed measurement model is used.

B. STEP AND HEADING EKF

We first visually compare the position estimates obtained with EKF-SH to those obtained with EKF-RW. For a fair comparison, we use the DD model in both EKF variations. We assume that the IMU magnetometer, or phone compass, is calibrated either through figure-8 calibration or through a software or firmware module. We make a few key observations in reference to Figure 10. First, EKF-SH trajectory estimates (purple) are better aligned with their ground truth (solid black line) than the EKF-RW trajectory estimates (blue). Second, the EKF-SH trajectory is smoother than the EKF-RW, which fluctuates a lot around the line of motion. Third, the EKF-SH trajectory shows discrete, and in many times equidistant dots corresponding to the regular steps that the user takes while walking.

Visual evaluation, though more indicative of the user-perceived experience, is not complete without numerical evaluation. We compare EKF-SH not only with EKF-RW, but also with other EKFs using step and heading motion models, namely (20) and (22), tagged "Ref. 1" and "Ref. 2" respectively. We evaluate the performance of the different filters through three commonly used metrics as introduced in Section V, the horizontal error (HE), cross-track error (XTE), and along-track error (ATE).

1) EKF-SH VERSUS EKF-RW

We plot the CDFs of the three errors in Figure 11 for EKF-SH and the baseline, EKF-RW, on Models A and B. We observe that EKF-SH outperforms EKF-RW across the entire curve through all three metrics, more so at the right end of the CDF curve corresponding to higher-valued errors, meaning that EKF-SH has a better worst-case accuracy than EKF-RW. If we measure performance through the 90th percentile error, EKF-SH achieves a gain of 30 cm in the HE over EKF-RW, most of which comes from a gain in the XTE.

2) EKF-SH VERSUS OTHER STEP-AND-HEADING BASED EKFS

In the same figure, we plot the CDFs of the three positioning error metrics for EKF-SH and two reference EKFs using motion models based on step and heading, namely Ref. 1 and Ref. 2 based on the motion equations (20) and (22). The EKF-SH beats Ref. 1 by 10 cm in the 90th percentile HE, 10 cm in ATE, and 20 cm in XTE. The EKF-SH beats Ref. 2 by a much wider margin across the board.

Measured on any device model or through any error metric, Ref. 1 sits somewhere between EKF-SH and EKF-RW, while Ref. 2 significantly underperforms EKF-RW, let alone EKF-SH. The reason behind the wide performance gap between Ref. 2 and Ref. 1, to the favor of the latter, is two-fold. First, two states in Ref. 2 are unobservable, the motion heading θ_k and the step size correction factor α_k, so they cannot be corrected by measurements of those states. Furthermore, there are no inputs to the transition equations of these two states, so they cannot be predicted. On the contrary, Ref. 1 has only one unobservable state, the motion heading θ_k, but this state transitions based on the rotation angle ϕ_k measured by sensors on the device, e.g. the gyroscope that measures rotational velocity, or magnetometer that measures absolute device orientation. This allows for predicting this state by reading relevant sensors.

Looking at the swaying range (SR) and rocking range (RR), we see that EKF-SH reduces the SR significantly, not so much, though, for RR.

Last but not least, a recurring observation is that positioning accuracy, regardless of the used metric, can vary by more than half a meter from one device model to another.

In summary, EKF-SH beats EKF-RW on all three counts of positioning errors: horizontal, cross-track, and along-track errors. This means three things. First, the EKF-SH estimates are closer to their ground truths than EKF-RW estimates. Second, EKF-SH estimates are closer to the track along which the user moves. Third, the EKF-SH estimates keep up with the user's walking and react faster to their latest position. Furthermore, EKF-SH reduces the jitter in position estimates both along and across the track of motion, making for a more smooth trajectory and a more pleasant user experience.

There are several factors that may explain why one device model outperforms the other. To begin with, the parameters of the FTM session influence ranging accuracy. The FTM mechanism, under the 11mc revision, allows the FTM initiator to request its desired configuration for the FTM session through the FTM parameters field. For example, the FTM initiator could request the number of bursts in a session, the duration of the burst, the number of measurements (FTM frames) per burst, the time between successive measurements, and the time between successive bursts. All of these parameters and more are hidden inside inaccessible, lower-level WiFi driver implementations, and it is believed that Models A and B

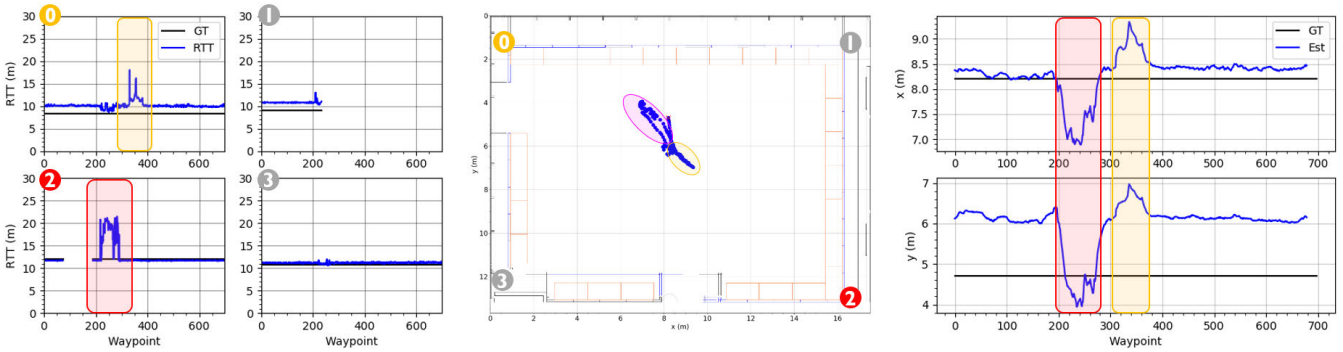


FIGURE 13. The human body blocks the signals carrying FTM frames, forcing them to take longer longer paths, and causing bigger ranges and farther position estimates; center figure: scatterplot of estimates of a static position marked by a red ×; left: 2 × 2 array of RTT distances vs. waypoints (time); right: *x* and *y* coordinates of the position estimate vs. waypoints.

have different FTM parameters. Moreover, Models A and B may be using different methods to calibrate their clock speeds which is critical for timestamping the transmission and reception of FTM frames and measuring distances. Finally, Models A and B have different WiFi chips and IMUs altogether, so there may be other variables at play.

VII. HUMAN BODY BLOCKAGES

We demonstrate that RTT is highly sensitive to the relative position of the human body holding the device measuring it. We present the following case study. A smartphone is mounted on a 1.5-m-high tripod at the center of a 17 m × 13 m empty square exhibition room with a 3-m-high AP at every corner as can be seen in Figure 13. The one person in the room stands close to the tripod, changing their position as intended. The experiment is divided into 4 segments: the first segment extends from $k = 0 \rightarrow 200$ (in waypoints, or equivalently observation indices), the second segment from $k = 200 \rightarrow 300$, the third from $k = 300 \rightarrow 400$, and the fourth and final segment extending till the end of the experiment.

A. FIRST SEGMENT

The user stands far from the tripod mounting the device, marked with a red ×. Measurements from all 4 APs are largely present, with the exception of a gap in measurements from AP 2. The ranges from AP 0 and AP 1 have a 1-2 m offset, resulting in about 1.1 m offset in the estimated position in the downward vertical direction. Offsets in range measurements push the position estimate away from its ground truth. Falling on the same horizontal line, APs 0 and 1 push the estimate to the right and to the left to similar degrees, effectively canceling out one another in the horizontal dimension. The two APs, however, both push the estimates downward, so the net effect the offsets in the ranges from the two APs have is an offset in position only in the downward vertical position.

B. SECOND SEGMENT

The user stands between AP 2 (red) and the tripod. As a result, the RTT distance jumps by 8 m from 12 m to 20 m, and remains at that level for the remainder of this segment. Furthermore, the position estimate moves away from AP 2 in the amount of 1 m to the right (negative *x*-axis) and 2 m to the top (negative *y*-direction). The corresponding position estimates are circled in a red oval. The reason the position moves not more than 3 m is the use of the distance-dependent model that gives little weight to the large RTT distances from AP 2 in the EKF update step and greater weights to RTT distances from the remaining 3 APs.

C. THIRD SEGMENT

The user now shifts their position to between the tripod and AP 0 (yellow). As a result, the RTT distance spikes by 5 m, falls back down, and spikes for a second time. This time around, the RTT distance does not stay high for the entire duration of the segment. The jump in RTT nudges the position estimate away from AP 0: 1 m to the left (positive *x*-axis) and 1m to the bottom (positive *y*-axis).

D. FINAL SEGMENT

The user now shifts their position to between the tripod and AP 1. Instead of observing a jump in RTT, the FTM exchanges between the device and AP 1 are lost. The position estimates are then produced using ranges from APs 0, 2, and 3. This outcome is more favorable than the outcomes previously seen: a missing observation can lead to a better estimate than a biased observation with large weight.

This experiment, among many other experiments we conducted, substantiates the fact that the position of the human body relative to the device and AP can have a detrimental impact on the quality of RTT. When the user holding the device is facing the AP, RTT fluctuates very little, and more importantly, has little bias. When the user's back is turned to the AP, RTT exhibits a large bias and pushes the position estimate away from the AP in question.

As the human body blocks the signal to and from the APs, the FTM frames reach the device through reflection off of the walls and diffraction around the user, taking multiple, longer paths to reach the destination and making for a longer RTT.

While the distance-dependent measurement model corrects a measured range in proportion to its value, a distance-direction model could correct a range in relation to both its

value and the direction of the user's motion with respect to the AP in question. This is deferred to future work.

VIII. CONCLUSION

In this paper, we described two positioning algorithms: the random walk extended Kalman filter (EKF-RW) which relies solely on RTT measurements, and the step and heading extended Kalman filter (EKF-SH) that supplements these measurements with sensor readings. We proposed two enhancements. Our first innovation is the distance-dependent measurement model used in either EKF, which computes the measurement noise mean and variance in the EKF update state as a function of the RTT measurements themselves. Our second innovation is a method to fit the non-Gaussian step error of the EKF-SH with a Gaussian random variable, which permits to uphold the efficient Kalman filtering framework. We benchmarked our algorithms and the enhancements thereof using three error metrics common to satellite and maritime navigation, the horizontal, cross-track, and along-track errors. We observed that the EKF-SH outperforms EKF-RW on all 3 counts and achieves a gain of up to 30 cm in the 90th percentile, and also that both comfortably beat the expected 2 m target at 80 MHz, making them primed for commercial deployment. Additionally, we introduced the swaying and rocking ranges to measure the jitter across and along the track of motion. We observed that EKF-SH achieves a gain of up to 60 cm over EKF-RW, making for a trajectory that is smoother and less choppy. Finally, we studied a case of human body blockage where we showed its detrimental impact on ranging errors, and as a consequence, on positioning errors.

The standardization of FTM under IEEE 802.11-2016 followed by the WFA's release of the *WiFi Location* certification sparked research on WiFi ranging as a serious contender to take over the IPS market, supported by the ubiquity of WiFi access points and stations alike. Research and industry continue to match the position resolution under WiFi RTT to the use case it is targeting, from the order of the size of a storefront at 40 MHz down to the order of aisle width at 80 MHz, prompting the IEEE to standardize enhanced ranging mechanisms and AP vendors to adopt them. Whether the next round of AP upgrades will support ranging over 160-MHz-wide channels unlocked by 802.11az *Next Generation Positioning* or over 320-MHz-wide channels unlocked by 802.11bk, indoor positioning is well on its to way to sub-meter and even centimeter-level accuracy, enabling a plethora of use cases in new domains.

APPENDIX

We arrived at a value for $\sigma_P \approx 3$ by taking a theoretical approach followed by an experimental validation. The random walk model assumes a directional step $\mathbf{v}_k$ that follows a circular Gaussian distribution $\mathcal{N}(0, \sigma_P^2 \Delta t_k^2)$ which is equivalent to a uniform velocity that follows a circular Gaussian distribution $\mathcal{N}(0, \sigma_P^2)$. A uniform velocity corresponds to a constant speed that is Rayleigh distributed with a parameter σ_P, a mean μ_s and a standard deviation σ_s, both of which are functions of σ_P.

We consider a speed range of 3 mph, or 1 m/s, for walking, to 7 mph, or 3.3 m/s, for running. We then fit the range by $\mu_s - \sigma_s$ on the slower end and $\mu_s - \sigma_s$ and on the faster end, and arrive at the solution that gives $\mu_s = 2.17$ and $\sigma_s = 1.13$. Finally, we solve backwards for the parameter σ_P from which the parameters μ_s and σ_s are derived and arrive at the value of $\sigma_P^2 = 3$.

Common positioning use cases, however, anticipate users to be walking at slower to moderate speeds. Therefore, a range of 1 mph, for strolling, to 5 mph, for jogging, would be more appropriate, and would correspond to a process noise variance $\sigma_P^2 \approx 1$. To see how well theory meets practice, we swept and evaluated a range of values for the process noise variance σ_P^2 on the 18 hours of data collected by 15 users with different walking patterns. We observed that, averaged across all datapoints, there is no significant difference in positioning accuracy for a σ_P^2 range of 1-3.

An alternative to a fixed σ_P is an adaptive, time-varying σ_P that infers the user's walking speed from the in-device accelerometer. But if acceleration is readily available, then more sophisticated, better performing PDR-based methods can be applied.

REFERENCES

[1] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2568–2599, 3rd Quart., 2019.

[2] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni, "A survey on wireless indoor localization from the device perspective," *ACM Comput. Surv.*, vol. 49, no. 2, pp. 1–31, Jun. 2017.

[3] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, R. Raulefs, and E. Aboutanios, "Recent advances in indoor localization: A survey on theoretical approaches and applications," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 1327–1346, 2nd Quart., 2017.

[4] Y. Hu, X. Fan, Z. Yin, F. Qian, Z. Ji, Y. Shu, Y. Han, Q. Xu, J. Liu, and P. Bahl, "The wisdom of 1,170 teams: Lessons and experiences from a large indoor localization competition," in *Proc. 29th Annu. Int. Conf. Mobile Comput. Netw. (MobiCom)*, Oct. 2023.

[5] Y. Hu, F. Qian, Z. Yin, Z. Li, Z. Ji, Y. Han, Q. Xu, and W. Jiang, "Experience: Practical indoor localization for malls," in *Proc. 28th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2022, pp. 82–93.

[6] D. Coppens, A. Shahid, S. Lemey, B. Van Herbruggen, C. Marshall, and E. De Poorter, "An overview of UWB standards and organizations (IEEE 802.15.4, FiRa, Apple): Interoperability aspects and future research directions," *IEEE Access*, vol. 10, pp. 70219–70241, 2022.

[7] C. Laoudias, A. Moreira, S. Kim, S. Lee, L. Wirola, and C. Fischione, "A survey of enabling technologies for network localization, tracking, and navigation," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 3607–3644, 4th Quart., 2018.

[8] P. Dabove, G. Ghinamo, and A. M. Lingua, "Inertial sensors for smartphones navigation," *SpringerPlus*, vol. 4, no. 1, p. 834, Dec. 2015.

[9] P. Davidson and R. Piché, "A survey of selected indoor positioning methods for smartphones," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 1347–1370, 2nd Quart., 2017.

[10] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.

[11] F. Potortì et al., "The IPIN 2019 indoor localisation competition—Description and results," *IEEE Access*, vol. 8, pp. 206674–206718, 2020.

[12] F. Potortì et al., "Off-line evaluation of indoor positioning systems in different scenarios: The experiences from IPIN 2020 competition," *IEEE Sensors J.*, vol. 22, no. 6, pp. 5011–5054, Mar. 2022.

[13] M. Orfanos, H. Perakis, V. Gikas, G. Retscher, T. Mpimis, I. Spyropoulou, and V. Papathanasopoulou, "Testing and evaluation of Wi-Fi RTT ranging technology for personal mobility applications," *Sensors*, vol. 23, no. 5, p. 2829, Mar. 2023.
[14] C. Ma, B. Wu, S. Poslad, and D. R. Selviah, "Wi-Fi RTT ranging performance characterization and positioning system design," *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 740–756, Feb. 2022.
[15] B. K. P. Horn, "Indoor localization using uncooperative Wi-Fi access points," *Sensors*, vol. 22, no. 8, p. 3091, Apr. 2022.
[16] J. A. López-Pastor, P. Arques-Lara, J. J. Franco-Peñaranda, A. J. García-Sánchez, and J. L. Gómez-Tornero, "Wi-Fi RTT-based active monopulse RADAR for single access point localization," *IEEE Access*, vol. 9, pp. 34755–34766, 2021.
[17] C. Gentner, M. Ulmschneider, I. Kuehner, and A. Dammann, "WiFi-RTT indoor positioning," in *Proc. IEEE/ION Position, Location Navigat. Symp. (PLANS)*, Apr. 2020, pp. 1029–1035.
[18] O. Hashem, M. Youssef, and K. A. Harras, "WiNar: RTT-based sub-meter indoor localization using commercial devices," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Mar. 2020, pp. 1–10.
[19] Y. Yu, R. Chen, Z. Liu, G. Guo, F. Ye, and L. Chen, "Wi-Fi fine time measurement: Data analysis and processing for indoor localisation," *J. Navigat.*, vol. 73, no. 5, pp. 1106–1128, Sep. 2020.
[20] G. Guo, R. Chen, F. Ye, X. Peng, Z. Liu, and Y. Pan, "Indoor smartphone localization: A hybrid WiFi RTT-RSS ranging approach," *IEEE Access*, vol. 7, pp. 176767–176781, 2019.
[21] M. Ibrahim, H. Liu, M. Jawahar, V. Nguyen, M. Gruteser, R. Howard, B. Yu, and F. Bai, "Verification: Accuracy evaluation of WiFi fine time measurements on an open platform," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2018, pp. 417–427.
[22] C. Li, L. Shi, N. Moayeri, and J. Benson, "A performance comparison of Wi-Fi RTT and UWB for RF ranging," Nat. Inst. Standards Technol. (NIST), Gaithersburg, MD, USA, 2019.
[23] S. Yan, H. Luo, F. Zhao, W. Shao, Z. Li, and A. Crivello, "Wi-Fi RTT based indoor positioning with dynamic weighted multidimensional scaling," in *Proc. Int. Conf. Indoor Positioning Indoor Navigat. (IPIN)*, Sep. 2019, pp. 1–8.
[24] H. Cao, Y. Wang, J. Bi, S. Xu, H. Qi, M. Si, and G. Yao, "WiFi RTT indoor positioning method based on Gaussian process regression for harsh environments," *IEEE Access*, vol. 8, pp. 215777–215786, 2020.
[25] P. Picazo-Martínez, C. Barroso-Fernández, J. Martín-Pérez, M. Groshev, and A. de la Oliva, "IEEE 802.11az indoor positioning with mmWave," 2023, *arXiv:2303.05996*.
[26] H. Cao, Y. Wang, J. Bi, Y. Zhang, G. Yao, Y. Feng, and M. Si, "LOS compensation and trusted NLOS recognition assisted WiFi RTT indoor positioning algorithm," *Expert Syst. Appl.*, vol. 243, Jun. 2024, Art. no. 122867.
[27] M. Sun, Y. Wang, S. Xu, H. Qi, and X. Hu, "Indoor positioning tightly coupled Wi-Fi FTM ranging and PDR based on the extended Kalman filter for smartphones," *IEEE Access*, vol. 8, pp. 49671–49684, 2020.
[28] J. Choi and Y.-S. Choi, "Calibration-free positioning technique using Wi-Fi ranging and built-in sensors of mobile devices," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 541–554, Jan. 2021.
[29] X. Liu, B. Zhou, P. Huang, W. Xue, Q. Li, J. Zhu, and L. Qiu, "Kalman filter-based data fusion of Wi-Fi RTT and PDR for indoor localization," *IEEE Sensors J.*, vol. 21, no. 6, pp. 8479–8490, Mar. 2021.
[30] M. Rea, D. Giustiniano, and J. Widmer, "Virtual inertial sensors with fine time measurements," in *Proc. Int. Conf. Mobile Ad Hoc Sensor Syst.*, Dec. 2020, pp. 658–665.
[31] S. Xu, R. Chen, Y. Yu, G. Guo, and L. Huang, "Locating smartphones indoors using built-in sensors and Wi-Fi ranging with an enhanced particle filter," *IEEE Access*, vol. 7, pp. 95140–95153, 2019.
[32] Y. Yu, R. Chen, L. Chen, G. Guo, F. Ye, and Z. Liu, "A robust dead reckoning algorithm based on Wi-Fi FTM and multiple sensors," *Remote Sens.*, vol. 11, no. 5, p. 504, Mar. 2019.
[33] K. Han, S. M. Yu, S.-L. Kim, and S.-W. Ko, "Exploiting user mobility for WiFi RTT positioning: A geometric approach," *IEEE Internet Things J.*, vol. 8, no. 19, pp. 14589–14606, Oct. 2021.
[34] B. Zhou, Z. Wu, Z. Chen, X. Liu, and Q. Li, "Wi-Fi RTT/encoder/INS-based robot indoor localization using smartphones," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6683–6694, May 2023.
[35] B. Carse, B. Meadows, R. Bowers, and P. Rowe, "Affordable clinical gait analysis: An assessment of the marker tracking accuracy of a new low-cost optical 3D motion analysis system," *Physiotherapy*, vol. 99, no. 4, pp. 347–351, Dec. 2013.
[36] P. Biswas, T.-C. Lian, T.-C. Wang, and Y. Ye, "Semidefinite programming based algorithms for sensor network localization," *ACM Trans. Sensor Netw.*, vol. 2, no. 2, pp. 188–220, May 2006.
[37] T. Jia and R. M. Buehrer, "A set-theoretic approach to collaborative position location for wireless networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 9, pp. 1264–1275, Sep. 2011.
[38] A. O. Hero and D. Blatt, "Sensor network source localization via projection onto convex sets (POCS)," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Mar. 2005, pp. 689–692.
[39] B. K. P. Horn, "Doubling the accuracy of indoor positioning: Frequency diversity," *Sensors*, vol. 20, no. 5, p. 1489, Mar. 2020.
[40] R. Harle, "A survey of indoor inertial positioning systems for pedestrians," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1281–1293, 3rd Quart., 2013.
[41] D. Laidig and T. Seel, "VQF: Highly accurate IMU orientation estimation with bias estimation and magnetic disturbance rejection," *Inf. Fusion*, vol. 91, pp. 187–204, Mar. 2023.
[42] H. Weinberg, "Using the ADXL202 in pedometer and personal navigation applications," Analog Devices, Wilmington, MA, USA, Tech. Rep. AN-602, 2002.
[43] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park, "A step, stride and heading determination for the pedestrian navigation system," *J. Global Positioning Syst.*, vol. 3, nos. 1–2, pp. 273–279, Dec. 2004.
[44] R. Chen, L. Pei, and Y. Chen, "A smart phone based PDR solution for indoor navigation," in *Proc. Int. Tech. Meeting Satell. Division Inst. Navigat.*, 2011, pp. 1404–1408.
[45] I. Klein and O. Asraf, "StepNet—Deep learning approaches for step length estimation," *IEEE Access*, vol. 8, pp. 85706–85713, 2020.
[46] O. Asraf, F. Shama, and I. Klein, "PDRNet: A deep-learning pedestrian dead reckoning framework," *IEEE Sensors J.*, vol. 22, no. 6, pp. 4932–4939, Mar. 2022.
[47] M. Alloulah, M. Arnold, and A. Isopoussu, "Deep inertial navigation using continuous domain adaptation and optimal transport," 2021, *arXiv:2106.15178*.
[48] C. Chen, P. Zhao, C. Xiaoxuan Lu, W. Wang, A. Markham, and N. Trigoni, "Deep learning based pedestrian inertial navigation: Methods, dataset and on-device inference," 2020, *arXiv:2001.04061*.
[49] H. Yan, S. Herath, and Y. Furukawa, "RoNIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, and new methods," 2019, *arXiv:1905.12853*.
[50] B. J. Odelson, A. Lutz, and J. B. Rawlings, "The autocovariance least-squares method for estimating covariances: Application to model-based control of chemical reactors," *IEEE Trans. Control Syst. Technol.*, vol. 14, no. 3, pp. 532–540, May 2006.

REBAL JURDI received the B.Eng. degree in computer and communications engineering from American University of Beirut, Lebanon, in 2013, and the M.S. and Ph.D. degrees in electrical and computer engineering from The University of Texas at Austin, in 2017 and 2019, respectively. He is currently a Staff Engineer with Samsung Research America. His research interests include wireless positioning and sensing.

HAO CHEN received the B.S. and M.S. degrees in information engineering from Xi'an Jiaotong University, Shaanxi, in 2010 and 2013, respectively, and the Ph.D. degree in electrical engineering from The University of Kansas, Lawrence, KS, USA, in 2017. Since 2016, he has been a Research Engineer with the Standards and Mobility Innovation Laboratory, Samsung Research America. His research interests include network optimization, machine learning, and 5G cellular systems.

YUMING ZHU (Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from Tsinghua University, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from The University of Arizona, in 2006. From 2006 to 2015, he was a System Architect with Research and Development Laboratories, Texas Instruments. He joined Samsung Research America, in 2015, and he is currently a Research Director. His current research interests include wireless communication, signal processing, RF sensing, and localization.

BOON LOONG NG received the Bachelor of Engineering degree in electrical and electronics and the Ph.D. degree in engineering from The University of Melbourne, Australia, in 2001 and 2007, respectively. He contributed to 3GPP RAN L1/L2 standardizations of LTE, LTE-A, LTE-A Pro, and 5G NR technologies, from 2008 to 2018. He holds more than 60 USPTO-granted patents on LTE/LTE-A/LTE-A Pro/5G and more than 100 patent applications globally. Since 2018, he has been leading the research and development team that develops system and algorithm design solutions for commercial 5G and Wi-Fi technologies. He currently holds the position of the Senior Research Director of the Standards and Mobility Innovation (SMI) Laboratory, Samsung Research America, Plano, TX, USA.

NEHA DAWAR received the B.Tech. degree in communication and computer engineering from The LNM Institute of Information Technology, Jaipur, India, in 2011, the M.S. degree in electrical engineering from the University of Calgary, Calgary, AB, Canada, in 2014, and the Ph.D. degree in electrical and computer engineering from The University of Texas at Dallas, Richardson, TX, USA, in 2018. She is currently with the Standards and Mobility Innovation (SMI) Laboratory, Samsung Research America (SRA), as a Staff Engineer. Her research interests include signal and image processing, wireless sensing, computer vision, and applied machine learning.

CHARLIE ZHANG received the Ph.D. degree from the University of Wisconsin-Madison. He was with the Nokia Research Center, from 2001 to 2006, and Motorola, from 2006 to 2007. From 2009 to 2013, he was the Vice Chairman of 3GPP RAN1 WG. He is currently the Senior Vice President and the Head of the Standards and Mobility Innovation Laboratory, Samsung Research America, where he leads research, prototyping, and standards for 5G cellular systems and future multimedia networks.

JOHNNY KYU-HUI HAN received the B.S. degree in computerized statistics from the University of Seoul, South Korea, in 2001. From 2016 to 2019, he achieved success in the development of Samsung Pay and mobile PoS based on NFC technology. He is currently a Principal Engineer (PE) with the Mobile Experience Division (MX), Samsung Electronics. His current research interests include indoor location positioning based on Wi-Fi, Bluetooth, and UWB technologies with his diverse work experience.