P2SLAM: Bearing Based WiFi SLAM for Indoor Robots

Aditya Arun, Roshan Ayyalasomayajula, William Hunter, and Dinesh Bharadia

***Abstract*—A recent spur of interest in indoor robotics has increased the importance of robust simultaneous localization and mapping algorithms in indoor scenarios. This robustness is typically provided by the use of multiple sensors which can correct each others' deficiencies. In this vein, exteroceptive sensors, like cameras and LiDAR's, employed for fusion are capable of correcting the drifts accumulated by wheel odometry or inertial measurement units (IMU's). However, these exteroceptive sensors are deficient in highly structured environments and dynamic lighting conditions. This letter will present WiFi as a robust and straightforward sensing modality capable of circumventing these issues. Specifically, we make three contributions. First, we will understand the necessary features to be extracted from WiFi signals. Second, we characterize the quality of these measurements. Third, we integrate these features with odometry into a state-of-art GraphSLAM backend. We present our results in a 25×30 m and 50×40 environment and robustly test the system by driving the robot a cumulative distance of over 1225 m in these two environments. We show an improvement of at least $6\times$ compared odometry-only estimation and perform on par with one of the state-of-the-art Visual-based SLAM.**

***Index Terms*—Sensor fusion, SLAM, localization.**

I. Introduction

DIVERSE applications including warehouse management, re-stocking supermarkets, and package delivery [1] have spurred research for indoor robotics, which extensively relies on visual sensors (cameras and LiDARs). To cater to these applications, indoor robotics use SLAM systems, wherein these visual sensors work in tandem with on-board odometry or IMU sensors. However, this broad spectrum of indoor applications brings unforeseen challenges for on-board visual sensors. Cameras provide low-cost, feature, and context-rich maps and location estimates but tend to fail in homogeneous feature-limited spaces like long corridors or bad-indoor lighting conditions [2]. Similarly, though LiDARs provide long-range, high-resolution, and relatively more descriptive sensing, they are too expensive and still suffer from limitations in long and monotonous corridors [3]. These limitations of the visual sensors motivate the inclusion of a low-cost and ubiquitous sensor in current robot SLAM frameworks.

Manuscript received September 9, 2021; accepted December 30, 2021. Date of publication January 25, 2022; date of current version February 8, 2022. This letter was recommended for publication by Associate Editor J. L. Blanco-Claraco and Editor J. Civera upon evaluation of the reviewers' comments.

The authors are with U.C. San Diego, La Jolla, CA 92093, USA (e-mail: aarun@eng.ucsd.edu; roshana@ucsd.edu; wshunter@ucsd.edu; dineshb@ucsd.edu).

Digital Object Identifier 10.1109/LRA.2022.3144796

We observe that Radio-frequency (RF) sensors are robust to dynamic lighting conditions and structured indoor environments and work even without visual line-of-sight [4]. Additionally, in indoor scenarios, WiFi is already ubiquitously deployed with many fixed anchors (APs) and boasts SWaP-C (power and cost) [5] metric $10\times$ better than LiDARs [6].[1] Thus, the objective we achieve in this letter is to demonstrate WiFi as a reliable and cost-effective SLAM sensor that requires no-loop closure, and can overcome the challenging indoor scenarios where cameras and LiDARs tend to fail.

A. Literature Review

Many RF-technology based SLAM implementations based on UWB [7], BLE [8], [9], RFID [10], or backscatter devices [11] exist. However, these RF technologies are not as ubiquitously deployed and have limited ranges compared to WiFi.

On the other hand, the majority of WiFi-based systems do not perform SLAM entirely using WiFi as a sensor. Instead, they create a localization and tracking system. It is assumed that the locations of the WiFi access points (APs) in the environment are known apriori, and the user or robot that carries the WiFi transmitter is localized to these already mapped APs [12] or vice-versa [13]. Thus, most of the existing systems use the location of either the robot or the AP to locate the other and cannot perform SLAM. In our method, P^2SLAM, we integrate WiFi as a sensor to simultaneously locate the robot and Map the WiFi APs in the environment.

Apart from the localization systems mentioned above, based on WiFi alone, there are a few existing works [14]–[17] that try to utilize WiFi sensor readings to improve SLAM algorithms. These works only try to use the WiFi signal strength (RSSI) features to predict 'loop closures' for camera-Lidar-based systems [14], [15] and not as an end-to-end SLAM sensor. A few works also use received signal strength (RSSI) features to integrate WiFi sensors, but these RSSI measurements are unreliable in indoor scenarios. It is well known that RSSI estimates vary drastically in dynamic indoor scenarios [12], making them unsuitable metrics for WiFi signals. In summary, a robust and end-to-end WiFi SLAM solution does not exist.

B. Objectives and Problem Statement

To overcome the limitations of visual sensors in challenging scenarios, we propose WiFi as an end-to-end, robust, and viable

[1] Due to space constraints further detailed results are presented along with the datasets at https://wcsng.ucsd.edu/p2slam

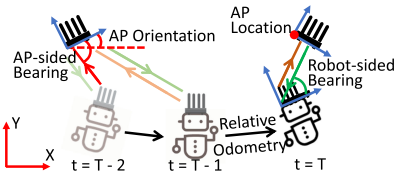

Fig. 1. P^2**SLAM:** The robots movement over three timestamp are shown. Relative odometry is measured between two consecutive robot poses. At each timestamp, the robot *ping's* an AP (orange arrow) and receives a *pong* reply (green arrow). For each ping and pong transmission the AP-sided and Robot-sided bearing of the signal is computed respectively.

sensor. The objective of P^2SLAM is to demonstrate the WiFi sensor's standalone performance as a SLAM sensor.

To develop WiFi as a reliable, low-cost, and easy-to-deploy SLAM sensor, let us see how the existing SLAM algorithms work with visual sensors. Most SLAM algorithms assume an onboard odometer or inertial sensor (IMU) alongside these sensors. While exploring, SLAM algorithms identify 'Landmarks' that correspond to unique features in an unknown environment. SLAM algorithms use these landmarks to track the robot and correct local drifts in odometry. Furthermore, these landmarks aid in 'loop closures' when re-identifying old landmarks to correct for global drifts, thus Simultaneously Localizing themselves and Mapping the environment (SLAM). So, to enable WiFi-based SLAM systems, we need to:

Identify 'Landmarks' and Unique Features: Firstly, identify the unique features to extract from the WiFi signals, ensure that they are unambiguous in structured and monotonous indoor environments. Furthermore, we need to extract these features without any prior knowledge about the environment (like the number or the locations of the access points).

Characterize the features: After identifying these features, we need to measure these unique features. Furthermore, we also need to model and characterize the noise and errors in these measurements to accurately integrate the features into SLAM frameworks.

Ensure Integrability and Scalability: Finally, we need these WiFi features to readily integrate with odometry measurements into openly available SLAM frameworks and scale to many indoor applications.

C. P^2*SLAM's Contributions*

To meet these objectives, in P^2SLAM we make the following three contributions that demonstrate WiFi as a standalone, reliable, low-cost, end-to-end SLAM sensor:

1) Landmark and Feature Identification: WiFi access points are widely deployed in indoor environments, making them reliable landmarks. However, P^2SLAM does not assume knowledge of the location of these access points, and thereof the environment.

Next, to identify the WiFi feature for SLAM integration, we make a note that WiFi systems perform to-and-fro transmissions between the WiFi access points (APs) and user or robot, as shown in Fig. 1. Thus, we can potentially have two distinct measurements/views from the 'ping'(AP to Robot, green arrows) and the 'pong' (Robot to AP, orange arrows). While most WiFi measurements like RSSI are reciprocal - the same information from ping and pong, we make a crucial observation that bearing relative to antenna array both at the AP and robot are distinct. Specifically, the 'ping' enables us to have the robot's pose relative to the AP's antenna array and vice-versa with the 'pong'. Both would be independent and distinct measurements. Thus, P^2SLAM identifies that two-way bearings are unique measurements that can be used as the WiFi features for a SLAM framework.

2) Feature Estimation and Characterization: Now that we have identified the 'two-way bearing' as our key feature, a natural next question for P^2SLAM is to estimate and characterize bearings relative to antenna arrays at both ends. The wireless signals reflect off objects in the environment, making estimating the signal's bearing challenging. Furthermore, we need to characterize the noise variance of the bearing measurement. Traditional algorithms use the MUSIC algorithm for bearing measurement to mitigate multi-path [12], but these algorithms are intractable to characterize the noise. P^2SLAM develops algorithms to measure the bearing with readily available channel state information from commercial-off-the-shelf WiFi devices while mitigating the multi-path effects. Furthermore, P^2SLAM provides a tangible approach to model the noise distribution of these two-way bearing measurements.

3) Feature Integration: Finally, these measurements need to be optimized over multiple poses of the robot in real-time and back-propagate any errors in these two-way bearing measurements to optimize for the overall robot, and AP poses. To accomplish this, P^2SLAM utilizes the open-source GraphSLAM library GTSAM [18], combining these two-way bearing measurements along with the relative-pose measurements from odometry within a factor graph. By optimizing this factor graph, we obtain accurate pose estimates of the robot and the WiFi APs in the environment.

Finally, we evaluate P^2SLAM in a 25×30 m and 50×40 m environment with 5 and 7 WiFi access points, respectively, which is a typical commercial deployment of WiFi anchors [19]. We use a Turtlebot2 [20] platform, equipped with a Hokuyo LiDAR [6], RGBD camera [21] and a WiFi transceiver [22]. We traverse a cumulative distance of over 1225 m in this environment to evaluate P^2SLAM thoroughly. We use Cartographer [23] to measure our ground truth. We observe a median (90^{th}) robot tracking accuracy of 26.9 cm (54.7 cm) and orientation accuracy of 1.28° (3.16°) for P^2SLAM, effectively showcasing the ability of WiFi radios as an end-to-end SLAM sensor.

II. WiFi as a Robust Sensor

For any new sensor to be integrated into the SLAM framework, we need to identify the sensor's measurements, estimate these measurements accurately and then characterize the errors in the estimation of the measurements. In P^2SLAM, we thus first identify in Section II-A that two-way bearing is the robust WiFi measurement that enables WiFi as a SLAM sensor. We then estimate and characterize the errors theoretically and verify them empirically in Section II-B

A. Robust and Reliable WiFi Features

To identify the features for our WiFi sensor, we first present the wireless propagation model employed for the WiFi 802.11ac protocol [19]. For simplicity, consider a single antenna at the transmitter (Tx) and multi-antenna receiver (Rx) on the robot as shown in Fig. 2(a). A signal broadcast from the Tx is received at the Rx as a combination of multiple paths, including the '*direct-path*' (green solid line) and the multiple reflected paths (green dotted lines). These multiple reflections are termed as '*multipath*'. WiFi signals in 802.11ac employ OFDM modulation wherein a series of N narrowband frequencies, or 'subcarriers,' are transmitted. So for the m^{th} antenna, n^{th} subcarrier, and K multipath components, we model the *raw CSI* or wireless channel $H^{m,n}$ as

$$H^{m,n} = \left[\underbrace{H_0^{m,n}(z_0, l_0)}_{\text{Direct Path}} + \underbrace{\sum_{i=1}^{K} H_i^{m,n}(z_i, l_i)}_{\text{Reflected Paths}} \right] e^{-j\phi}$$

$$H_i^{m,n} = a_i . e^{-j\left(2\pi \frac{f_0}{c}(m-1)d \sin z_i\right)} e^{-j\left(2\pi \frac{f_n}{c} l_i\right)} \quad (1)$$

with a_i as the amplitude, z_i and l_i as the angle of arrival and path length of each incoming signal at the robot, respectively. Furthermore, ϕ accounts for random phase offsets and timing offsets introduced due to lack of synchronization between the access points and the robot [12].

To narrow down on the features to utilize from the measured raw CSI, let us look at various components that can be extracted from the model in (1). A typically measured quantity at the m^{th} received antenna is the received signal strength measurement, $\text{RSSI}^m = \sum_n |H^{m,n}|^2$. Unfortunately, limited knowledge of the direct path and varying reflected paths means we cannot trust our RSSI measurements [12]. On the other hand, one can use the complete Channel State Information (CSI) given by $H^{m,n}\ \forall m, n$ in Eq (1). Unfortunately, characterizing this *raw* CSI would require modeling the signal amplitude (a_i), the bearing (z_i) and path-lengths (l_i) for K paths and the random phase offset (ϕ). Considering P different robot poses, at each pose, p, the number of paths in the wireless channel is K_p. Hence, this would require optimizing up to $\sum_{p=1}^{P}(3K_p + 1)$ parameters. Clearly, we cannot use the raw CSI as the WiFi measurement.

Our crucial insight is extracting the physical features like the angle of arrival (bearing, z_0) or path length (range, l_0), for the direct path, H_0, from the 'raw-CSI,' H, can be reliable and robust WiFi features. Additionally, WiFi standards ensure that the WiFi-client (our robot in this case) broadcasts a *ping* to all the AP's in the environment, which then reply with a *pong* to the WiFi client (our robot), hence identifying these AP's which behave as the fixed 'Landmarks'. Furthermore, this ping-pong exchange (the namesake of our work; P^2SLAM stands for Ping-Pong SLAM) allows us to measure the bearing of the signal independently at the robot and the AP each time a ping-packet is broadcast from the robot. These pair of measurements, the two-way bearings, are unique from the robot to the AP and vice-versa. Alternatively, we have the choice to measure the path length traversed by these signals [24]. However, these range measurements are degenerate; both the robot and the AP provide the same measurement robbing us of the opportunity to constrain the optimizer better.

With these considerations, we conclude 'two-sided bearings' are the ideal measurements. To estimate these bearings either at the bot or the AP in an ideal scenario with no reflectors ($H^{m,n} = H_0^{m,n}$), can be done as $\arcsin\left(\frac{(\angle H^{m,1} - \angle H^{1,1})c}{2\pi f_0 (m-1) d}\right)$. Specifically, we measure the bearings at either the robot or the AP is estimated from the phase difference between 1^{st} and m^{th} antenna. Unfortunately, multipath issues limit accurate estimations of these bearing values.

B. Two-Way Bearing Estimation & Characterization

To make our bearing estimates unaffected by multipath and work in real-time, we utilize 2D-FFT-based bearing estimation algorithms described in [4]. There are other algorithms like MUSIC and SpotFi that perform multipath rejection as well [25], but these algorithms are intractable to characterize the noise. For simplicity, we will begin the 2D-FFT-based bearing estimate by ignoring the AWGN noise in the channel model described in Section II-A. Given the *raw* CSI at the m^{th} receiver antenna at frequency f_n as $\hat{H}_{m,n}$, we generate a 2D 'likelihood' profile by taking the element-wise absolute of $\hat{\Lambda} \in \mathbb{C}^{|Z| \times |L|}$. This profile is generated by scanning over a set of possible bearings ($Z = \{z_i | z_{min} < z < z_{max}\}$) and distance measurements ($L = \{l | l_{min} < l < l_{max})\}$). An element of this profile, Λ_{ij} is generated using the following equation for the given bearing z_i and distance l_j,

$$\alpha_m(z) = e^{j\frac{2\pi f_0}{c}(m-1)d \sin z_i}; \quad \tau_n(l) = e^{j\frac{2\pi f_n}{c} l_j} \quad (2)$$

$$\hat{\Lambda}_{ij} = \sum_{n=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{m=1}^{M} \hat{H}_{n,m} \overbrace{\alpha_m(z_i)}^{\text{Bearing Phase}} \overbrace{\tau_n(l_j)}^{\text{ToF Phase}} \quad (3)$$

given channel measurements across N frequency bins centered around center-frequency f_0 for M antennas on the receiver. Thus, to identify the correct bearing for a given robot pose at a given WiFi anchor, we can compute all the local maxima in $\hat{\Lambda}$ and choose the maxima with the least l_j value, corresponding to the direct path. This bearing measurement corresponds to the direct path's bearing ($\hat{z}$). Given the reciprocal nature of channel measurement [19], we can utilize the same estimation technique for both the bearings on the robot ($\hat{z}_p^{\text{robot}}$) and the AP($\hat{z}_p^{\text{AP}}$) for a robot position p. Furthermore, having identified the direct path's bearing, we reconstruct a *cleaned* CSI as $\hat{H}^{\text{direct}} = H_0(z_0 = \hat{z}, l_0 = 0)$. We set l_0 to zero as it does not affect the bearing estimation.

Rejecting outlier bearing measurements: Regardless of multipath rejection, the bearing estimates are erroneous when a metal reflector blocks the direct path. We observe that the RSSI measurements are low in these scenarios, so we filter them out using an RSSI threshold. Hence, accounting for multipath and filtering out non-line of sight measurements gives us two-way bearing measurements for the WiFi sensor, which can be modeled and plugged into SLAM frameworks.

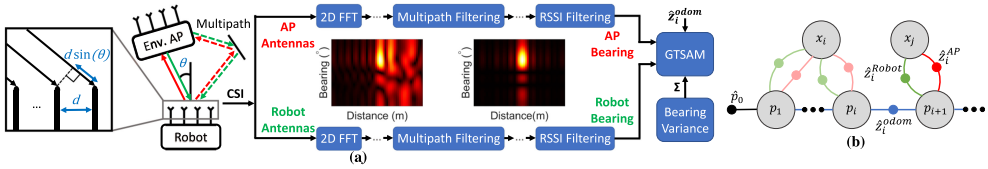

Fig. 2. **(a) P^{2}SLAM's Design Overview:** Via a ping and pong packet exchange, the robot collects the wireless channel (CSI) on both the AP and robot end. Performing a Fast Fourier Transform (FFT) and filtering out multi-paths from these channels effectively extracts the bearing at the AP and robot. RSSI filtering helps to remove outlier measurements. These 2-way bearing measurements, along with odometry measurements, and their measurement covariances, are fed into a SLAM framework. **(b) P^{2}SLAM's Factor Graph:** Visualization of factors and their connections once we include the two-way bearing measurements as the WiFi features in P^{2}SLAM's implementation of GraphSLAM.

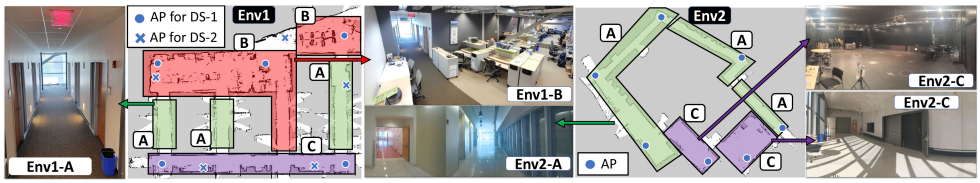

Fig. 3. Shows the two environments: For DS-1 and DS-2 collected in 25×30m for Env1 with different AP placement shown in blue circles and crosses, respectively, in the 2D map. For DS-3 collected in 40×50m for Env2 with AP placements shown as blue circles in the 2D map. Photos for the spaces are classified as (A) Office space, (B) Long corridors, and (C) Dynamic lighting spaces.

Finally, our goal is to incorporate the bearing measurements into GraphSLAM, which requires modeling the noise of the bearings. We develop for the first time the mathematical model to characterize the variance in the bearing errors due to the underlying AWGN in the CSI measurements. In order to model the variance, let us consider only the direct path's cleaned CSI matrix corrupted only with AWGN, $\hat{H}^{\text{direct}} \in \mathbb{C}^{M \times N}$ measured across M antennas and N subcarriers. From this CSI matrix, the direct path's likelihood profile $\hat{\Lambda}^{\text{direct}}$ can be computed using Eq (2) as shown in second profile in Fig. 2(a). Since we have only considered the direct path component, there will only be one maxima in $\hat{\Lambda}^{\text{direct}}$ at the estimated bearing ($\hat{z}$) and path-length $l_0 = 0$. Hence, $\hat{z} = \text{argmax}_{z_i}(\hat{\Lambda}_i^{direct})^2$.

Further, without loss of generality, we can model the AWGN for each of the elements of the matrix $\hat{H}_i^{direct} \sim \mathcal{N}(H_i, \sigma^2)$ as independent and identically distributed (IID) complex circularly symmetric additive Gaussian noise [26]. Under this assumption, the distribution of $\hat{\Lambda}_i$ is a complex Gaussian, with the real and complex parts independently distributed as $\hat{\Lambda}_i^{direct} \sim \mathcal{N}\left(\sum_{n=-\frac{-N}{2}}^{\frac{N}{2}} \alpha_m(z_i) H_n,\ \sigma^2_{\Lambda,i}\right)$, where, $\sigma^2_{\Lambda,i} = \sum_{n=-\frac{-N}{2}}^{\frac{N}{2}} |\alpha_m(z_i)|^2 \sigma^2 = N\sigma^2$.

Next, we take the squared magnitude of this complex Gaussian distribution which yields a scaled non-central χ^2_2 distribution [27], of $(\hat{\Lambda}_{ij})^2 = \sigma^2_{\Lambda,i} \kappa_{ij}$, where

$$\kappa_{ij} \sim \chi^2_2 \left(\lambda = \left(\sum_{n=-\frac{-N}{2}}^{\frac{N}{2}} \alpha_n(z_i) H_n \right)^2 \right)$$

where λ is the non-centrality parameter of the distribution. Next, we ravel the two indices $(i, j) \rightarrow k$ and compute the probability of k being maximum index, k_{max}, amongst all the candidate bearing values. Following the technique in [28],

$$\Pr(k_{max} = k) = \int_0^\infty \left[\prod_{p=1, p \neq k}^{|Z||L|} F_p\left(\frac{x}{\sigma^2_{\Lambda,p}} \right) \right] f_k\left(\frac{x}{\sigma^2_{\Lambda,p}} \right) dx$$

where F_k and f_k are the distribution and density functions of κ_k [29]. With this model, when we plug in the variance, σ^2, in our CSI measurements, we get a theoretical variance estimate which we have verified with our empirical measurements by measuring the bearing errors observed in real-world experiments. We thus use the two-way bearing as features and their characterized variance in the GraphSLAM framework.

III. Putting It All Together

P^{2}SLAM's primary design principle is to demonstrate WiFi as a robust, low-cost, and readily integrable SLAM sensor. To demonstrate that in P^{2}SLAM, we integrate the features we have identified and characterized into GraphSLAM [30].

A. Primer: Incorporating Wheel Odometry

To begin, we assume to have the readings from the wheel odometry of the robot. Wheel odometry and gyroscope sensors provide relative pose measurements across time steps. The state space, S, is a set of robot poses over P time steps in our graph, $S = \{\vec{p}_i \mid \vec{p}_i = (p_i^x, p_i^y, p_i^\theta) \in SE(2),\ \forall i \in [1, P]\}$, where each p_i belongs to the Special Euclidean group ($SE(2)$). Our relative pose measurement is z_i with a diagonal covariance matrix $\Sigma_i \in \mathbb{R}^3$. The covariance matrix is estimated by empirically modeling the noise in the odometer and gyroscope beforehand. To define the measurement model $\hat{z}^{odom}(\cdot)$, we consider two consecutive poses $\vec{p}_i$ and $\vec{p}_{i+1}$. The measurement model is

$$\hat{z}^{odom}(\vec{p}_i, \vec{p}_{i+1}) = \begin{bmatrix} R(-p_i^\theta)[p_{i+1}^x - p_i^x,\ p_{i+1}^y - p_i^y]^T \\ (p_{i+1}^\theta - p_i^\theta) \end{bmatrix}$$

where, $R(\cdot) \in SO(2)$ is the rotation matrix. This prediction function now allows us to constrain two consecutive robot poses via a between factor within GraphSLAM [18] and defines an error function between the measurement and prediction as $e_i^{odom}(p_i, p_{i+1}) = z_i^{odom} - \hat{z}_i^{odom}(p_i, p_{i+1})$ as shown by the blue edges in Fig. 2(b).

B. Incorporating P^{2}SLAM's 2-Way Bearings

With the odometry factors added, we turn our attention to incorporating the two-sided bearing measurements within P^{2}SLAM's factor graph. As shown in Fig. 2(b), we utilize two bearing factors – 'green' for the robot-sided bearings and 'red' for the AP-sided bearing. Next, we use a Between Factor (in 'blue') to incorporate the $\hat{zodom}_i$. In the following section, we will discuss the construction of this factor graph.

First, P^{2}SLAM does not assume the pose of these access points, hence we append the poses of Q WiFi APs (x) to our state space. The complete state space would then become $\{p_i \; \forall i \in [1, P]\} \bigcup \{x_j \; \forall j \in [1, Q]\}$. We thus redefine our prediction function in (4) to obtain the angle subtended by access point x_j at robot pose p_i. With this redefinition, P^{2}SLAM brings the poses of the access point into the purview of the optimization, accounting for the error in its prediction.

$$\hat{z}_{ij}^{\text{robot}}(p_i, x_j) = R(-p_i^{\theta})\text{atan2}(x_i^x - p_i^x, x_i^y - p_i^y) \quad (4)$$

where $\text{atan2}(\cdot)$ is the "2-argument" inverse tangent function. The rotation $R(-p_i^{\theta})$ brings the bearing of the access point into the local coordinate frame of the robot. This bearing measurement is made when the WiFi radio receives the 'pong' acknowledgement from the WiFi access points.

Similarly, we incorporate the feedback from the environment via 'ping' requests received at the access point. This measurement provides the robot's bearing in the AP's local frame of reference. Specifically, P^{2}SLAM obtains the bearing angle subtended in the local frame of reference at AP j when the robot is in the i^{th} position with the prediction function $\hat{z}_{ij}^{AP}(p_i, x_j) = R(-x_j^{\theta})\text{atan2}(x_i^x - p_i^x, x_i^y - p_i^y)$.

Furthermore, AP's are not always placed at the same height as the robot. More often than not, they are placed near the ceiling, and AP's height is also unknown apriori and adds to the complexity of the optimization. However, we make an important observation that would allow P^{2}SLAM to function in these scenarios – the bearings measured at the AP (and conversely at the robot) vary little with a change in the AP's height and negligibly affect the performance of P^{2}SLAM as elaborate in Section V-B4

Finally, to anchor the robot to a fixed global frame of reference and align it with the ground truth trajectory and map generated by Cartographer [23], we further provide an initial prior, $\hat{p}_0 \in \text{SE}(2)$, to the first pose. The error in this factor is $e_0 = [[p_0^x - \hat{p}_0^x,\; p_0^y - \hat{p}_0^y]^T, abs(p_0^{\theta} - \hat{p}_i^{\theta})]^T$.

With the factor graph constructed as shown in Fig. 2(b), we define the complete optimization function defined as,

$$S_{opt} = \text{argmin}_S \sum_{i=1}^{P}\sum_{j=1}^{Q} \rho\left(\frac{\left(e_{ij}^{AP}\right)^2}{\sigma^{AP}}; c\right) + \sum_{i=1}^{P}\sum_{j=1}^{Q} \rho\left(\frac{\left(e_{ij}^{Robot}\right)^2}{\sigma^{AP}}; c\right) + \sum_{i=1}^{P} \left(e_i^{odom}\right)^T \Sigma_i^{-1} e_i^{odom}. \quad (5)$$

where $e_{ij}^{bearing} = z_{ij}^{bearing} - \hat{z}^{bearing}(p_i, x_j)$ is the bearing factor's error between robot or AP poses i and j. To further strengthen the optimization to outliers, we use a Huber cost function, $\rho(\cdot, c)$ [31], for which we empirically find the threshold $c = 1.345$ provides the best estimate. This function is easily minimized in linear time proportional to the number of nodes initialized $N + M$, via Levenberg-Marquadt [18].

The final puzzle piece is to initialize each of the P robot positions and Q AP positions for the optimizer. We initialize the robot positions using the relative odometry measurements. Note that these priors will be affected by drift accumulated by the odometer. Additionally, we do not assume any prior location information for the AP's in the environment and initialize all the locations and orientations of the Q APs at the origin oriented towards the global X-axis.

Thus in P^{2}SLAM, we integrate WiFi into a SLAM framework by identifying and characterizing two-way bearing measurements as the sensory features. We then integrate these two-way bearings into the widely utilized and open-sourced state-of-the-art GraphSLAM implementation GTSAM [18]. We achieve all this without any prior knowledge or mapping available for any WiFi sensors in this unknown environment.

IV. Implementation

To demonstrate how P^{2}SLAM makes WiFi a low-cost, robust, and end-to-end SLAM sensor, we build a robot and deploy it in real-world environments as described below.

Robot: We build our robot to test P^{2}SLAM on the Turtlebot2 platform [20]. For our implementation of a WiFi sensor, we affix an off-the-shelf Quantenna WiFi transmitter [22] onto the Turtlebot2 via a raised acrylic platform at the height of one meter off of the ground. We transmit the WiFi packets according to the IEEE 802.11ac protocol [19], using channel 36 at 80 MHz bandwidth. We control the Turtlebot2 via Robot Operating System (ROS-Kinetic) running on a laptop equipped with an 8th Gen Intel Core i5-8250 U Processor and 8 GB of RAM that collects all the data and implements P^{2}SLAM. The Turtlebot 2 also comes equipped with a wheel encoder and gyroscope, which we utilize for dead reckoning. The pose-graph optimization and bearing estimation are performed on a 12-core Intel i7-6800 K CPU. We collect LiDAR scan data from a Hokuyo UTM-30LX LiDAR mounted on the robot at the height of 50 cm. We also mount an Intel D415 Camera below the Hokuyo LiDAR to collect RGB-D camera data [21] for visual SLAM. Ground truth information is collected from Cartographer [23] using the LiDAR and wheel-odometry on board the Turtlebot2. We use the open-sourced RTABMap [32] implementation for visual SLAM and further fine-tune for benchmarking.

Environment: We deploy P^{2}SLAM in two different environments (Env1 and Env2). Across these two environments, we cover three different scenarios, a few of which are challenging to WiFi signals and a few for camera-based systems. We classify them into three broad categories as labeled in Fig. 3: (A)

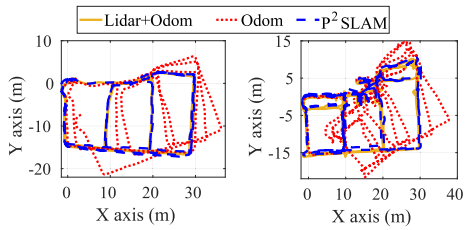

Fig. 4. **Top-down views** of P^{2}SLAM's optimized path compared against the ground truth from Cartographer (yellow, *LiDAR+Odom*) and odometry-only dead-reckoning (red, *Odom*). (*Left*) Dataset-1 (*Right*) Dataset-2. Note the large drifts accumulated by odometry effectively corrected by P^{2}SLAM.

Monotonous Corridors: (In green) Long monotonous corridors, where the camera features are more ambiguous and narrow corridors that cause heavy multipath for WiFi signals, (B) *Office Spaces:* (In red) Typical office spaces with cubicles, screens, metal cabinets that cause most of the multipath for the indoor environments for the WiFi, while providing unique camera features for Visual SLAM algorithms, and (C) *Dynamic Lighting:* (In Purple) These are the spaces either right next to glass panes facing outdoors, and closed spaces with low lighting conditions creating more dynamic lighting conditions that act as adversaries for the camera features. Each photo highlights an example section of each type in the map. This figure is best viewed in a colored format of this letter. Next, in Env1 (Env2), we leverage 5 (7) Quantenna access points [22] already deployed in a 25×30 m (50×40 m) environment, a standard deployment density for indoor environments [19]. We traverse this environment with the above robot platform in two different scenarios. In Dataset-1 (DS1) collected in Env1, the robot traverses the various corridors at an average velocity of 18 cm/s for a total distance of 332.6 m. In Dataset-2 (DS2) collected in Env1, the robot makes various tight turns and follows a more natural path typical for a robot delivering packages or mapping an environment. In Dataset-2, we traverse a total path length as long as 496 m at an average velocity of 28 cm/s. In Dataset-3 (DS3) collected in Env2, the robot goes through many long narrow corridors and changing lighting conditions and completes 2.5 loops around the entire space. In Dataset-3, we traverse a total path length as long as 400 m at an average velocity of 28 cm/s.

V. Evaluation

We evaluate P^{2}SLAM in the three datasets described in Section IV, where the robot has traversed a total of more than 1225 m across DS1/2/3. We compare our results from P^{2}SLAM's implementation with one of the open-sourced state-of-the-art Visual SLAM algorithms, RTABMap [32]. We see that in the cases where there are dynamic lighting conditions and under lack of completion of loops P^{2}SLAM's translation and orientation are better than RTABMap's and elsewhere P^{2}SLAM performs on-par with the state-of-the-art Visual SLAM algorithm in estimating the bots poses when the AP locations are unknown as well.

We evaluate P^{2}SLAM against the LiDAR-based Cartographer ground truth. We acknowledge that this LiDAR-based odometry may also be erroneous, but rigorous testing [4] has found these errors to be within 15 cm at the 90^{th} percentile. The translation error is computed as the square root of the L2 errors (RMSE) and orientation errors L1 errors in the Special Euclidean group (SO(2)). The errors are computed between the predictions from P^{2}SLAM, RTABMap, or odometry-only dead-reckoning and the Cartographer ground-truth.

TABLE I
Comparison of Translation and Orientation Error for All the Datasets. The Median Errors and 90^{th} Percentile Errors (in Parentheses) are Shown

Algorithm	Env 1-Dataset 1		Env 1-Dataset 2		Env 2-Dataset 3	
	Trans [cm]	Orient [°]	Trans [cm]	Orient [°]	Trans [cm]	Orient [°]
Dead-reckoning	180.6 (513.9)	8.64 (18.1)	378.6 (1156)	23.35 (37)	422 (1098)	16 (30.5)
RTABMap [32]	36.8 (165.7)	2.97 (10.83)	**38.5** (**63.7**)	**0.74** (**2.69**)	**61.5** (256)	2.2 (7.99)
P2SLAM	**26.9** (**54.7**)	**1.28** (**3.16**)	40.4 (76.9)	1.32 (3.7)	65.2 (**158**)	**1.65** (**3.95**)

Best performing metrics for each column.

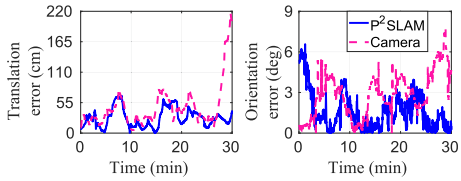

Fig. 5. **Timeseries of errors for Dataset-1:** (*Left*) Timeseries of translation error, (*Right*) Timeseries of robot orientation error.

A. End-to-End Evaluation

Firstly, we evaluate the end-to-end performance of P^{2}SLAM and compare it with state-of-the-art Camera-sensor implementation of RTABMap and the default Odometry-only dead-reckoning estimates of the Turtlebot2. We evaluate it in two scenarios and report the median and 90^{th} percentile errors for both the translation and orientation errors in Table I. From this table, we can see that P^{2}SLAM outperforms Odometry-only dead-reckoning across all three datasets by at least $7\times$ ($10\times$) in the median (90^{th} percentile) translation errors. Further, across all the three datasets P^{2}SLAM outperforms Odometry only dead-reckoning by $10\times$ ($10\times$) in the median (90^{th} percentile) orientation errors. We further show a qualitative output of the Lidar ground truth compared with the P^{2}SLAM's and Odometry-only dead reckoning's trajectories in Fig. 4 to show that P^{2}SLAM corrects for these crude errors by integrating two-way bearings from WiFi sensor.

From the Table I, we can observe that P^{2}SLAM performs on-par with RTABMap in terms of median and 90^{th} percentile for orientation and translation error across all the three datasets. We can also see that P^{2}SLAM outperforms RTABMap at both the median and 90^{th} percentiles for both the orientation and translation errors for Dataset-1 and Dataset-3. We want to draw attention here to the 90^{th} percentile of RTABMap, which is at least $3\times$ ($2\times$) worse for both the Translation and Orientation errors compared to the median errors that are comparable for Dataset-1 (Dataset-3). These errors are from RTABMap's incorrect loop-closures in the last 5 minutes of the robot's exploration. This is further seen from Fig. 5 for Dataset-1, where we can see

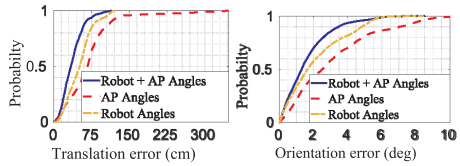

Fig. 6. **Need for Two-Way Bearings:** Comparison between the use of just the access point-sided bearings, the robot-sided bearings, and both of them simultaneously. (*Left*) comparative CDF of translation errors, (*Right*) comparative CDF of orientation errors.

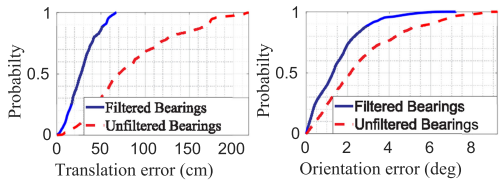

Fig. 7. **Filtering measurements:** Showcases the importance of filtering out the measurements based on RSSI and the measured bearing. (*Left*) comparative CDF of translation errors, (*Right*) comparative CDF of orientation errors.

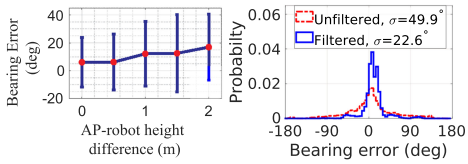

Fig. 8. **Microbenchmarks:** (*Left*) Robot and AP-sided bearing errors (median and standard deviation shown in error bars) when the AP is placed at different heights. (*Right*) Histogram of unfiltered and filtered bearing measurements' errors. Note the reduction in standard deviation, indicating the removal of erroneous measurements.

that RTABMap loses tracking at the 25 min mark because it encounters a sudden glare. Finally, we can see from Table I that P^{2}SLAM performs on par with RTABMap in Dataset-2 as the Camera sensor has no adverse scenarios.[2]

We can thus see that P^{2}SLAM's two-way bearings based WiFi integration into the GraphSLAM framework enables WiFi as an ideal sensor for indoor scenarios where the conditions are adversarial to the camera and LiDAR sensors like long corridors and dynamic lighting conditions.

B. Ablation Studies

Next, we perform some ablation studies to understand the importance of individual components of P^{2}SLAM's design.

1) One-Way vs. Two-Way Bearing: Firstly, the core assumption of P^{2}SLAM's WiFi measurement as claimed in Section II-A is to utilize two-way bearings over only one-way bearings. The intuition behind this was that the two-way bearings would provide both local and global feedback for each robot's pose, enabling simultaneous optimization of robot and AP poses. To evaluate this intuition, we turn off each of the 'ping' and 'pong' bearing in the factor graph and compare it with the two-way bearing-based implementation and show the obtained results as a Translation error and Orientation error CDFs in Fig. 6.

In Fig. 6 (Left), we show the CDF of the translation error after optimizing with the bearing measurements at both the AP and robot, at only the AP and only the robot. We see a 34% and 49% improvement at the median when using two-sided bearings instead of only using the robot-sided or AP-sided bearings. Furthermore, the bearing measurements at the robot are affected by the robot's orientation, whereas those at the AP are not. Hence, measurements at the robot carry additional information about the robot's orientation. This additional benefit is apparent in the orientation error CDF in Fig. 6 (Right), as incorporating bearing at the robot improves the orientation error by 50% at the median as opposed to only using the bearing measurements at the AP.

Finally, we note that the time to compute the bearing for a single packet is on average 12 ms. Furthermore, using only the environment's AP-sided bearing or only the robot-sided bearing, the optimization takes 94.2 sec and 237.03 sec respectively, for a factor graph containing 26150 factors. Whereas, using two-sided bearings, end-to-end optimization takes 8.549 sec, demonstrating the efficacy of two-sided bearings.

2) RSSI Filtering of Measurements: As mentioned in Section II-B, we use RSSI-based filtering to reject non-line-of-sight WiFi measurements. We observe higher bearing errors in these measurements, which detrimentally impacts the navigation performance. Furthermore, we find that bearing errors for measurements outside the $\pm 60^\circ$ range is higher. We show the effects of these filtering techniques in the CDFs of translation and orientation errors in Fig. 7. This RSSI-based and measurement-based filtering helps to exclude the erroneous measurements collected, reducing the translation (orientation) error by 58% (37%). The exclusion of these outlier measurements reduces the variance of errors, which is shown to be an essential factor affecting accuracy [33]. Specifically, we observe that the RSSI and measured bearing-based filtering reduces the standard deviation of the error distribution from 49.9° to 22.6° as shown in Fig. 8(right). Finally, these reduced variances are used in the GraphSLAM framework described in Section II-B.

TABLE II
RSSI VALUES FOR EACH OF THE SEVEN AP'S USED ACROSS THE TWO ENVIRONMENTS. NOTE THAT DS1/2 USE ONLY THE FIRST 5 AP'S (1-5)

AP	1	2	3	4	5	6	7
RSSI (dBm)	-40	-35	-35	-35	-40	-35	-35

3) Generalizability of RSSI Measurements: One of the major concerns for RSSI-based filtering is that these RSSI thresholds applied are environment-specific. To ensure reproducibility, we deployed P^{2}SLAM in two environments. We found that the performance remains the same across these two environments for the same RSSI thresholds for all the APs as shown in Table II. We observe that the RSSI is a proxy for the *hardware noise*, an environment-independent, AP-specific parameter. The AP reports this hardware noise and can serve as a one-time calibration.

4) Effect of AP's Height on Measurement: We have experimented with varying the AP height between 1 m and 3 m. The robot subtends angles between -60° to 60° at the AP. As mentioned in Section IV, the robot's antennas are placed at 1 m. We observe the median bearing errors at different heights changing from 6.1° to 16.9° as shown in Fig. 8 (Left), which is a tolerable error for P^{2}SLAM.

Hence, we show the importance of two-way bearings to improve the final accuracy and RSSI filtering to reject outliers, the generalizability of these RSSI thresholds, and the effect of the AP's height on the bearing measurements.

VI. Future Work and Limitations

In P^2SLAM, we have identified, characterized, and integrated two-way bearing measurements as features to develop a WiFi-based sensor front-end for existing GraphSLAM architectures. With this idea, we have implemented P^2SLAM on open-sourced GraphSLAM implementation [18] framework. The current implementation of P^2SLAM is a first step to show the feasibility of WiFi as a sensor that can easily replace expensive LiDAR's and augment visual sensors in challenging situations of poor lighting and featureless scenarios. However, WiFi-based sensors have limitations, and future work is needed to integrate them into SLAM robots.

P^2SLAM provides a robust and inexpensive solution for indoor navigation using WiFi sensors. Nevertheless, it cannot provide visual and contextual information about dynamic obstacles in the environment to prevent collisions. Additionally, we have observed Doppler effects introduced for bot speeds over 50 cm/s affect the accuracy of bearing measurements. We also note that P^2SLAM's implementation can be extended to any AoA-based wireless sensors (UWB [34] or BLE [35]). Finally, future work to provide ROS support to WiFi-based sensors [22] would improve the accessibility of these sensors.

Acknowledgment

The authors would like to take this opportunity to thank all the anonymous reviwers, members of WCSNG Lab, and Nikolay A. Atanasov for all their insightful comments and feedback

References

[1] T. Deyle, "Why indoor robots for commercial spaces are the next big thing in robotics," *IEEE Spectr.*, 2017, Accessed: Oct. 31, 2020. [Online]. Available: https://spectrum.ieee.org/indoor-robots-for-commercial-spaces

[2] X. Deng, Z. Zhang, A. Sintov, J. Huang, and T. Bretl, "Feature-constrained active visual slam for mobile robot navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 7233–7238.

[3] A. Zhang and M. M. Atia, "Comparison of 2D localization using radar and LiDAR in long corridors," in *Proc. IEEE Sensors*, 2020, pp. 1–4.

[4] R. Ayyalasomayajula *et al.*, "Deep learning based wireless localization for indoor navigation," in *Proc. 26th Annu. Int. Conf. Mobile Comput. Netw.*, 2020, pp. 1–14.

[5] R. C. Allen, W. B. Blanton, E. Schramm, and R. Mitra, "Strategies for reducing SWAP-C and complexity in DVE sensor systems," *Proc. SPIE*, 2017, vol. 10197, Art. no. 101970.

[6] " P. Demski, M. Mikulski, and R. Koteras, "Characterization of hokuyoutm-30lx laser range finder for an autonomous mobile robot," in *Adv. Tech. Intell. Syst. Nat. Border Secur.*, Springer, 2013, pp. 143–153.

[7] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "VIRALSLAM: tightly coupled camera-imu-uwb-lidar SLAM," *CoRR*, vol.abs/2105.03296, 2021. [Online]. Available: https://arxiv.org/abs/2105.03296.

[8] A. Sato, M. Nakajima, and N. Kohtake, "Rapid BLE beacon localization with range-only EKF-SLAM using beacon interval constraint," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigation*, 2019, pp. 1–8.

[9] M. G. Jadidi, M. Patel, J. V. Miro, G. Dissanayake, J. Biehl, and A. Girgensohn, "A radio-inertial localization and tracking system with BLE beacons prior maps," in *Proc. IEEE Int. Conf. Indoor Positioning Indoor Navigation*, 2018, pp. 206–212.

[10] Y. Ma, N. Selby, and F. Adib, "Drone relays for battery-free networks," in *Proc. Conf. ACM Special Int. Group Data Commun.*, 2017, pp. 335–347.

[11] S. Zhang, W. Wang, S. Tang, S. Jin, and T. Jiang, "Robot-assisted backscatter localization for IoT applications," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 5807–5818, Sep. 2020.

[12] Y. Ma, G. Zhou, and S. Wang, "WiFi sensing with channel state information: A survey," *ACM Comput. Surv.*, vol. 52, no. 3, pp. 1–36, 2019.

[13] R. Ayyalasomayajula *et al.*, "LocAP: Autonomous millimeter accurate mapping of WiFi infrastructure," in *Proc. 17th {USENIX} Symp. Networked Syst. Des. Implementation*, 2020, pp. 1115–1129.

[14] Z. S. Hashemifar, C. Adhivarahan, A. Balakrishnan, and K. Dantu, "Augmenting visual SLAM with Wi-Fi sensing for indoor applications," *Auton. Robots*, vol. 43, no. 8, pp. 2245–2260, 2019.

[15] R. Liu *et al.*, "Collaborative SLAM based on WiFi fingerprint similarity and motion information," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1826–1840, Mar. 2020.

[16] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor WiFi graphSLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2011, pp. 1038–1043.

[17] B. Ferris, D. Fox, and N. D. Lawrence, "WiFi-SLAM using Gaussian process latent variable models," *Int. Joint Conf. Artif. Intell.*, vol. 7, no. 1, pp. 2480–2485, 2007.

[18] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the Bayes tree," *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 216–235, 2012.

[19] E. Perahia and R. Stacey, *Next Generation Wireless LANs: 802.11 N and 802.11 Ac*. Cambridge, U.K.: Cambridge Univ. Press, 2013.

[20] "Turtlebot 2," *Turtlebot Robot.*, Accessed: Jan. 31, 2022. [Online]. Available: https://www.turtlebot.com/turtlebot2/

[21] Intel, "RealSense D415," Accessed: Jan. 31, 2022. [Online]. Available: https://www.intelrealsense.com/depth-camera-d415/

[22] M. Rea, T. E. Abrudan, D. Giustiniano, H. Claussen, and V.-M. Kolmonen, "Smartphone positioning with radio measurements from a single wifiaccess point," in *Proc. 15th Int. Conf. Emerg. Netw. Exp. Tech.*, 2019, pp. 200–206.

[23] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 1271–1278.

[24] M. Ibrahim *et al.*, "Verification: Accuracy evaluation of WiFi fine time measurements on an open platform," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, 2018, pp. 417–427.

[25] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "SpotFi: Decimeter level localization using Wi-Fi," in *Proc. ACM Conf. Special Int. Group Data Commun.*, 2015, pp. 269–282.

[26] N. R. Goodman, "Statistical analysis based on a certain multivariate complex gaussian distribution (an introduction)," *Ann. Math. Statist.*, vol. 34, no. 1, pp. 152–177, 1963.

[27] K. H. Biyari and W. C. Lindsey, "Statistical distributions of hermitian quadratic forms in complex Gaussian variables," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 1076–1082, May 1993.

[28] S. Soltani and M. W. Mutka, "ArgMax and ArgMin: Transitional probabilistic models in cognitive radio mesh networks," *Wireless Commun. Mobile Comput.*, vol. 15, no. 9, pp. 1355–1367, 2015.

[29] A. Stuart, J. K. Ord, and S. Arnold, "Kendall's advanced theory of statistics. vol. 2a, Classical inference and the linear model," *Kendall's Adv. Theory Statist. Vol. 2 A, Classical Inference Linear Model*, 6th ed. New York: Oxford Univ. Press, 1999, p. 865.

[30] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intell. Transp. Syst. Mag.*, vol. 2, no. 4, pp. 31–43, Winter, 2010.

[31] Z. Zhang, "Parameter estimation techniques: A tutorial with application to conic fitting," *Image Vis. Comput.*, vol. 15, no. 1, pp. 59–76, 1997.

[32] M. Labbé, "RTAB-Map as an open-source LiDAR and visual SLAM library for large-scale and long-term online operation," *J. Field Robot.*, vol. 36, no. 2, pp. 416–446, 2019.

[33] C. Cadena *et al.*, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1309–1332, Dec. 2016.

[34] M. Zhao, T. Chang, A. Arun, R. Ayyalasomayajula, C. Zhang, and D. Bharadia, "ULoc: Low-power, scalable and cm-accurate UWB-tag localization and tracking for indoor applications," *Proc. ACM Interactive, Mobile, Wearable Ubiquitous Technol.*, 2021, vol. 5, no. 3, pp. 1–31.

[35] R. Ayyalasomayajula, D. Vasisht, and D. Bharadia, "BLoc: CSI-based accurate localization for BLE tags," in *Proc. 14th Int. Conf. Emerg. Netw. Experiments Technol.*, 2018, pp. 126–138.