Robot-assisted Backscatter Localization for IoT Applications

Shengkai Zhang, *Student Member, IEEE,* Wei Wang*, *Senior Member, IEEE,* Sheyang Tang, Shi Jin, *Member, IEEE,* and Tao Jiang, *Fellow, IEEE*

***Abstract*—Recent years have witnessed the rapid proliferation of backscatter technologies that realize the ubiquitous and long-term connectivity to empower smart cities and smart homes. Localizing such backscatter tags is crucial for IoT-based smart applications. However, current backscatter localization systems require prior knowledge of the site, either a map or landmarks with known positions, which is laborious for deployment. To empower universal localization service, this paper presents Rover, an indoor localization system that localizes multiple backscatter tags without any start-up cost using a robot equipped with inertial sensors. Rover runs in a joint optimization framework, fusing measurements from backscattered WiFi signals and inertial sensors to simultaneously estimate the locations of both the robot and the connected tags. Our design addresses practical issues including interference among multiple tags, real-time processing, as well as the data marginalization problem in dealing with degenerated motions. We prototype Rover using off-the-shelf WiFi chips and customized backscatter tags. Our experiments show that Rover achieves localization accuracies of** 39.3 **cm for the robot and** 74.6 **cm for the tags.**

***Index Terms*—Backscatter, localization, inertial sensor, channel state information**

I. Introduction

In the last few years, the rapid innovations of small-footprint and low-power backscatters in both end-to-end communication [2]–[7] and large-scale networking [8]–[10] have been driving the realization of the universal Internet-of-Things (IoT) deployment. Their designs enable concurrent communications among a large number of IoT devices and low-power communications that avoid the inconvenience of changing the battery. Localizing such universal backscatters is crucial for ubiquitous sensing and smart services in both domestic and industrial fields. For example, the location of a dog chew toy can be monitored for the convenience of home cleaning and the RF penetration capability from backscatter tags enables the item tracking in highly cluttered settings for industrial production, *e.g.*, tracking an item from under a pile [11] to help a robot pick it.

Part of this work has been presented at IEEE GLOBECOM [1].

This work was supported in part by the National Key R&D Program of China under Grant 2019YFB180003400, Young Elite Scientists Sponsorship Program by CAST under Grant 2018QNRC001, National Science Foundation of China with Grant 91738202.

S. Zhang, W. Wang, S. Tang and T. Jiang are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China. (E-mail: {szhangk, weiwangw, sheyangtang, taojiang}@hust.edu.cn).

S. Jin is with the School of Information Science and Engineering, Southeast University, Nanjing 211189, China. (E-mail: jinshi@seu.edu.cn).

*The corresponding author is Wei Wang (weiwangw@hust.edu.cn).

To date, the fundamental challenge of backscatter localization has been its limited communication range due to the low-power signals, which turns out that the localizability is limited in a small region, *e.g.*, a room [12], or requires dense landmark deployment [11] that increases the start-up cost. To overcome this challenge, LoRa's high sensitivity holds the opportunity to enable long-range sensing with extremely low-power signals [13] and the superior mobility of drones [14] breaks the spatial limitation by approaching the tags and magnify the backscattered signals. Unfortunately, these works either require dedicated RF sources which are not ubiquitously available [13], or need additional sensing modalities to obtain the location of the vehicle first [14].

Ideally, we desire a system that supports the IoT localization with low-power signals that extends the tracking demand from smartphones and wearables to universal objects, such as wallets, keys, and lost items: the system should be power-on-and-go that works without any effort for start-up, *e.g.*, site survey or landmark position setup, and it should be ubiquitously available and lightweight that works with low-cost sensor suite upon existing infrastructure for rapid deployment.

In this paper, we propose Rover, a power-on-and-go backscatter localization system that works with existing ubiquitous infrastructure, *i.e.*, commodity WiFi, and supports instant deployment with zero start-up cost. The CSI of multiple subcarriers of WiFi packets holds the fine-grained localizability [15]. It is a self-contained system that runs on a robot equipped with WiFi chips and an inertial measurement unit (IMU). Rover simultaneously localizes the robot and the backscatter tags that communicate with the robot. It needs to rove in the work space to connect to more tags for localization. Rover addresses the range challenge with the mobility of a robot, who approaches the tags and localizes them by a simultaneous-localization-and-mapping (SLAM) solution. It leverages spatially different observations to construct multi-view constraints for localizing the tags and the robot.

Despite the advantages of Rover, there are three significant challenges. *First*, without knowing the positions of a robot or any tag, enough translations and angle-of-arrivals (AoAs) of a tag to the robot at different positions are required to satisfy the requirement of triangulation. A straightforward method to obtain the translation is integrating the accelerations measured by IMU. However, the integration operation will lead to a temporal drift of results due to the inherent sensor noise [16]. To address this issue, we exploit the drift-free localizability via WiFi AoAs by triangulation. We correct the IMU drift by proposing an AoA-IMU SLAM system that jointly optimizes locations of the robot and the connected backscatter tags with WiFi AoAs and IMU measurements.

Second, the optimization framework takes the measurements from WiFi and IMU at different locations and our goal is to find a configuration of such locations that best fit all these measurement constraints. In principle, taking more measurements over the robot's trajectory into account for the optimization would provide more accurate results. However, this also incurs more complex computations and delays the localization, making the robot unable to navigate itself when moving. To bound the computation complexity for real-time processing while achieving high accuracy, Rover employs a sliding window based formulation for the SLAM problem and derives the solution in a graph-based optimization framework.

Third, the sliding window based formulation for the SLAM problem involves a marginalization operation that removes old states in the window when obtaining new observations. A simple data marginalization scheme is first-in-first-out (FIFO). However, FIFO cannot handle degenerate motions, *e.g.*, being stationary or moving at a constant velocity, as in this case the data in the window cannot recover the metric scale of environments. We propose a flexible marginalization scheme to address this issue.

Results. We prototype Rover on the programmable robot, iRobot Create 2, equipped with an IMU and an Intel Next Unit of Computing (NUC) that installs an Intel 5300 wireless NIC attached with three antennas. We use the 802.11n Channel State Information (CSI) tool [17] to obtain wireless channel information for AoA estimation. To implement the backscatter tag, we use the hardware provided by HitchHike [18] and reprogram its FPGA with our Rover firmware. The experiments are conducted with four backscatters deployed in a conference room of our laboratory to validate individual system modules as well as the overall performance. The results show that Rover is capable of handling the robot's degenerated motions and achieves localization accuracies of 74.6 cm for the backscatter tag and 39.3 cm for the robot over a trajectory of 41.96 m.

Contributions. Rover is the first backscatter localization system that works with a single robot using commodity WiFi without any prior knowledge of work space. Rover leverages the localizability of WiFi signals to correct the IMU drift. *First*, we propose an AoA-IMU SLAM system that jointly optimizes the locations of the robot and the connected backscatter tags subject to measurement constraints of IMU and WiFi. *Second*, we employ a sliding window based formulation for the SLAM problem to achieve real-time processing. *Third*, we devise a flexible marginalization scheme for the sliding window operation to handle degenerated motions. We implement Rover on commodity devices and experimentally validate the system in indoor environments.

The rest of this paper is divided into four parts. Section II introduces the interference avoidance scheme and the AoA estimation technique. Section III presents the sliding window based AoA-IMU SLAM system as well as the flexible marginalization scheme. We show the results of system evaluations in Section IV and summarize the related works in Section V. Finally, Section VI concludes this work and points out possible directions for future improvements.

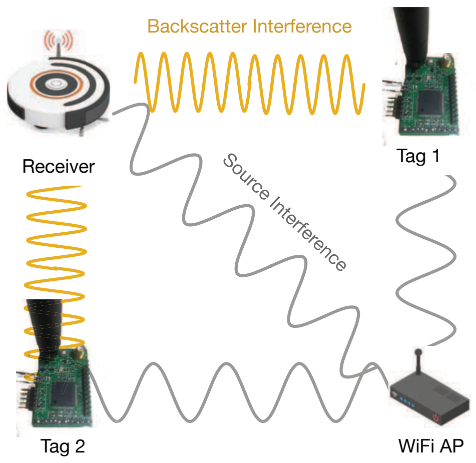

Fig. 1. The interference between the backscattered signals (yellow wave) from tags and the excitation signal (grey wave) from a transmitter.

II. Backscatter AoA Estimation

Differing from conventional WiFi localization systems that the target device responds to a WiFi receiver through an active WiFi radio, backscatter localization systems arise two problems: 1) The WiFi receiver should be able to receive and decode the backscattered low-power signals from tags; 2) Multiple tags that concurrently backscatter signals can interfere with each other or with the WiFi transmitter as shown in Fig. 1. The key to address the first problem is to make sure the tag reflects the preamble of the excitation WiFi packet. We implement an envelop detector to detect the starting point of a WiFi packet and backscatter a decodable WiFi packet as proposed in [12]. To resolve the interference, frequency shifting [18] is effective that we can build a tag that shifts the WiFi signal by a particular frequency to another channel and then backscatters the frequency-shifted signal. Multiple tags need to shift into different channels. While the number of WiFi channels is limited, we propose an interference avoidance mechanism that allows Rover to work in a scalable battery-free network.

A. Interference Avoidance

To avoid the interference from a WiFi transmitter, a straightforward solution is to move the backscattered signal to another channel that does not overlap with the channel where the excitation WiFi signal is sent. We achieve this by toggling its RF transistor at a higher speed [18], *e.g.*, 20 MHz. Then, the backscattered signal will be moved to a channel that is 20 MHz away from the channel where the excitation WiFi signal stays. Configuring the receiver to work on the channel of the backscattered signal will address the interference from the excitation signal.

To avoid the interference from multiple tags that concurrently backscatter signals, the above idea can be extended to assign different available WiFi channels to each of the tags by toggling their RF transistors to different speeds. In this way, the tags can be distinguished by their allocated WiFi channels. Each channel corresponds to a particular tag. To

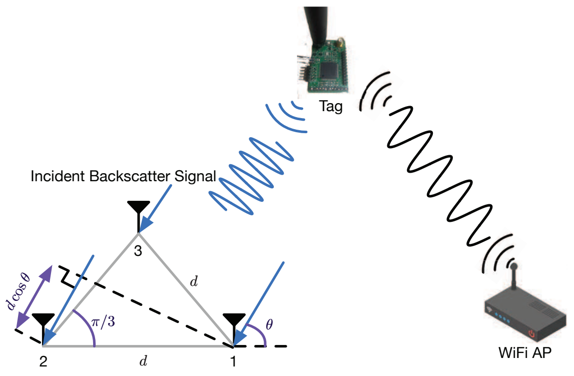

Fig. 2. A uniform circular array that consists of three antennas in our system: The signal with AoA θ travels an additional distance of $d\cos\left(\theta+\frac{\pi}{3}\right)$ to the third antenna and $d\cos(\theta)$ to the second array in the array compared to the first antenna.

receive packets from all tags, the receiver in our system needs to be capable of sweeping all WiFi channels except the channel of the excitation signal. We achieve this by implementing a frequency band sweeping protocol [19] in the iwlwifi driver of Intel 5300 NIC. Since the number of non-overlapped WiFi bands is limited, Rover can only simultaneously localize a limited number of tags. To maximize the ability of simultaneous localization, it is vital to choose the channel of the excitation signal.

Suppose a tag uses a frequency f_b square wave signal to control the on-off frequency of the RF switch. f_c is the carrier center frequency of the 802.11n excitation signal. Let $\omega_b = 2\pi f_b$, $\omega_c = 2\pi f_c$, and $\alpha_{\text{base}}(t)$ denotes a baseband waveform. The square wave can be formulated as $S_{\text{tag}} = \frac{4}{\pi}\sum_{n=1}^{\infty}\frac{\sin[(2n-1)\omega_b t]}{2n-1}$. Hence, the backscattered signal $\beta(t)$ can be written as,

$$\beta(t) = \alpha_{\text{base}}(t)e^{j\omega_c t}S_{\text{tag}}(t). \tag{1}$$

Let $F_{\text{base}}(\omega)$ and $F(\omega)$ be the Fourier transform of $\alpha_{\text{base}}(t)$ and $\beta(t)$ respectively. We have

$$\begin{aligned} F(\omega) = \sum_{n=1}^{\infty}\frac{2j}{\pi(2n-1)}\left(F_{\text{base}}\left(\omega-\omega_c+(2n-1)\omega_b\right)-\right. \\ \left.F_{\text{base}}\left(\omega-\omega_c-(2n-1)\omega_b\right)\right). \end{aligned} \tag{2}$$

This indicates that the frequency-shifted backscattered signal can be received at two bands, $f_c \pm f_b$, causing sideband interference to other channels. Based on this, Rover chooses the most side channels in the band, *i.e.*, channel 165 or 36, to transmit the the excitation signal in order to avoid the sideband interference.

B. AoA Estimation

So far, we have addressed the interference problem. A WiFi receiver can receive the backscatter packets from tags. In this section, we describe how Rover estimates the AoAs of backscatter tags to the receiver, which is amounted on a robot, leveraging the CSI of received packets. The AoA estimation technique for low-power backscatter tags was first proposed in [12]. Here we extend it to work with a circular antenna array with uniform spacing d that can measure AoAs in $[0, 360]$ degrees as shown in Fig. 2.

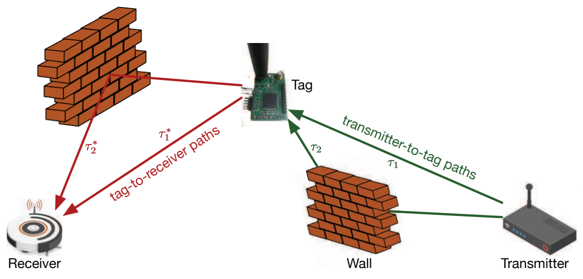

Fig. 3. The received signal traverses two physical paths where τ_j denotes the time of flight (ToF) of j^{th}path on the transmitter-to-tag link and τ_i^* denotes the traversing time of i^{th}path on the tag-to-receiver link.

Localizing backscatter tags involves two physical paths, transmitter-to-tag path and tag-to-receiver path, as shown in Fig. 3. Thus, the received CSI depends on the locations of backscattered tags and the access point (AP). We combine j^{th}path on the transmitter-to-tag link with i^{th}path on the tag-to-receiver link to form a virtual path between the excitation source (AP) and the receiver at the robot. The virtual path has a ToF of $\widehat{\tau}_k = \tau_j + \tau_i^*$ where τ_i^* (τ_j) denotes the ToF of the signal along a path on the tag-to-receiver (transmitter-to-tag) link, the AoA of the virtual path $\widehat{\theta}_k = \theta_i^*$ where θ_i^* is the AoA of i^{th}path on the tag-to-receiver link, and the corresponding complex attenuation of $\widehat{\gamma}_k = \gamma_j\gamma_i^*$ where γ_i^* and γ_j denote the complex attenuation along i^{th}path on the tag-to-receiver link and j^{th}path on the transmitter-to-tag link, respectively. The overall signal obtained at the three antennas for n^{th}subcarrier can be written as

$$\begin{aligned} H_{n,m} &= \sum_{k=1}^{L_{\text{tx}}L_{\text{tag}}}\widehat{\gamma}_k e^{-j2\pi\left(\widehat{\tau}_k(n-1)f_\delta+(m-1)d\cos\widehat{\theta}_k/\lambda\right)}, m=1,2 \\ H_{n,3} &= \sum_{k=1}^{L_{\text{tx}}L_{\text{tag}}}\widehat{\gamma}_k e^{-j2\pi\left(\widehat{\tau}_k(n-1)f_\delta+d\cos\left(\widehat{\theta}_k+\frac{\pi}{3}\right)/\lambda\right)}, \end{aligned} \tag{3}$$

where L_{tag} is the number of paths on the tag-to-receiver link, L_{tx} is the number of paths on the transmitter-to-tag link, f_δ is the frequency gap between two consecutive subcarriers. This overall signal is reported as CSI corresponding to the particular subcarrier and antenna.

The signal model is a standard form to apply a joint AoA-ToF estimation technique [20]. The insight of this technique is that multiple subcarriers of an OFDM signal encode ToF information. By smoothing the subcarriers represented in the CSI matrix, it allows a super-resolution AoA estimation with a small antenna array, *e.g.*, a three-antenna array available for Intel 5300 NIC, jointly estimating AoAs and ToFs[1] of all paths. The AoA of the path with the smallest ToF is the is the direct-path AoA of a tag to the receiver.

Unfortunately, the obtained AoA can be corrupted by the heading direction of the robot. The robot has three degrees of freedom, including 2D position and the heading direction.

[1]This ToF cannot correctly infer the traveling distance of a propagation path due to its poor distance resolution from the narrowband signal.

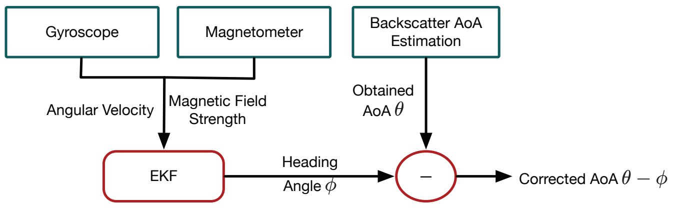

Fig. 4. The workflow of AoA correction. We first use the extended Kalman filter to fuse the magnetic field strength with the angular velocity to correct the drift of gyroscope. Then we can trivially correct the AoA.

It can rotate and change its heading direction while moving, *e.g.*, turning at a corner of the room. Thus, the onboard antenna array will be turning with the robot together. The system can no longer use the measured AoA to localize the robot via triangulation because the AoA not only encodes the geometric constraint of translations but also manifests the rotation (refer to Fig. 5 for details). Therefore, we need to correct the rotation from the measured AoA and recover the angle that only relates to the translation.

Fig. 4 shows the workflow of AoA correction. Basically, we leverage the IMU to estimate the robot's heading direction in angle ϕ, assuming that the initial heading direction is angle $0°$. The gyroscope in IMU provides raw measurements of angular velocity. However, it is well-known that simply integrating them to obtain the heading will result in error accumulation. We correct the heading by using a magnetometer onboard that provides a reference direction represented by the magnetic field strength. Note that in indoor venues, the reference direction is not the earth's North due to the magnetic interference from surrounding electronic devices. Nevertheless, the reference direction is stable in few hours so that it is eligible to correct the drift during the trajectory. Since the rotation estimation is non-linear, we employ the extended Kalman filter (EKF) to determine the heading by fusing the measurements from gyroscope and magnetometer [21]. Finally, we correct the AoA by subtracting the heading angle ϕ from the obtained AoA θ.

At this stage, we obtain the corrected direct-path AoAs of multiple tags to the receiver. Next, we fuse them with the IMU measurements to localize the tags and the robot simultaneously.

III. SLAM with AoAs

In this section, we first describe the design of our AoA-IMU localization system and then elaborating on the sliding window based formulation. Then, we propose a flexible marginalization scheme for addressing a practical issue of sliding window operations in degenerated motions.

A. AoA-IMU Localization System

The angle (AoA) can be used to determine a target's location via triangulation. Recall that conventional localization systems usually require a few landmarks with known locations to localize the target. The essence of this requirement is defining the metric scale of environments, *i.e.*, the unit (meter, millimeter, etc.) in measuring distances between objects, to fix the size of triangles.

In Rover, since the location of both the tags and the robot are unknown, the AoA we obtained cannot yield locations with the metric scale of environments. Nevertheless, with the aid of the onboard IMU and the mobility of a robot, we can localize both the connected tags and the robot in that the IMU provides accelerations in unit m/s^2, defining the metric scale. With the translations and the AoAs of incident signals at different positions, it forms a fixed triangle. We take one tag as an example illustrated in Fig. 5. As a robot moves, the IMU measures translation Δd and the antenna array measures AoAs θ_1 and θ_2 referring to the tag at different positions, one can determine the relative positions of the robot and the tag through triangulation. Note that the measured AoA has to be corrected from the rotation ϕ so as to obtain the correct geometric constraint between the tag and the robot.

Obtaining the translation by integrating the accelerations from the IMU is straightforward but suffers from temporal accumulated errors due to the inherent noise [16], causing large localization errors once the result severely distorts the triangle in Fig. 5. To address this issue, we develop an AoA-IMU SLAM approach that optimizes the locations of the robot and backscatters subject to measurement constraints with respect to WiFi AoAs and the IMU odometry.

Roughly speaking, the central idea of SLAM is to obtain a maximum likelihood estimate of both robot positions and environment features (backscatter tags in our system) given observations (AoAs) from the antenna array. Solutions to the SLAM problem can be either filtering-based or graph-based approaches. While filtering-based approaches are considered to be more efficient in computation [22], we choose graph-based approaches that can achieve better performance via repetitively linearizing past robot states and multi-view constraints [23].

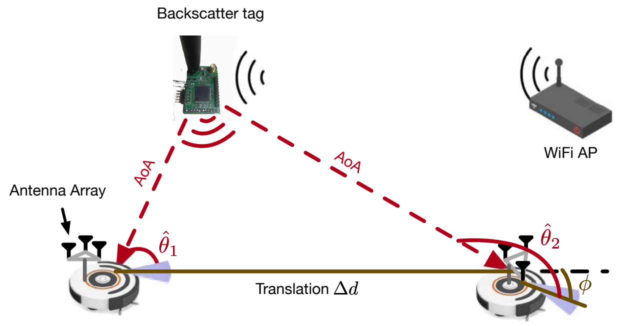

Fig. 5. Localization principle: triangulation with the robot's motions. The AP sends WiFi packets to excite the backscatter tag. The receiver on the robot measures the AoAs of the tag to the robot from backscatter signals and the onboard IMU measures translation Δd to provide the metric scale of environments. $\hat{(\cdot)}$ denotes the measured AoA and ϕ is the rotation of the robot since the previous state.

In addition, solving the SLAM problem is a batch process that incorporates multiple observations to produce accurate results. However, it can become unacceptably slow as the size of the environment grows. This delays the location estimates of the robot so that the robot loses its own navigation capability, being unable to move along the desired trajectory. To let our system run in real-time, we employ an incremental update method to speed up the computation. We formulate a sliding window based model that only keeps a limited amount of AoAs and corresponding robot hidden *states*, *e.g.*, the positions of the robot at different timestamps in the workspace, to bound the computation complexity.

B. System Overview

Upon the introduction of the AoA-IMU localization principle, we give an overview of our system as shown in Fig. 6. Basically, Rover uses two sensors: the Intel 5300 WiFi NIC as an exteroceptive sensor that observes the AoAs with respect to backscatter tags and the IMU as an interoceptive sensor that observes the dynamics of the robot. The sensor data are buffered in a sliding window array for bounding the computation complexity. Then the SLAM-based system model takes the data to solve out the locations of the tags as well as the robot.

Intuitively, the AoA observed by the NIC imposes a geometrical constraint that imply the relative locations of the robot and the connected tags. Note that only the corrected AoA (refer to Fig. 4) manifests the constraint. To localize the tags, we need to move the robot and capture the dynamics of the robot by IMU. The IMU provides the odometry constraint that indicates the locations of the robot by integrating angular rates and accelerations. Although it is well known that such an integration suffers from a temporal drift, the drift can be corrected by combining the AoA constraint (refer to Section III-C).

To limit the states and observations in the sliding window, we need to marginalize the data in the window when new observations come. A vanilla option of data marginalization is first-in-first-out (FIFO) that marginalizes out the oldest state and its corresponding measurements. This however cannot handle degenerate motions, *e.g.*, being stationary or moving at a constant velocity. Specifically, if a robot stays at a position for a moment, the measurements keep updating and rendering the sliding window so that the data in the window are all related to the same position. This cannot correctly recover the metric scale by triangulation because the translation Δd (Fig. 5) almost diminishes. If a robot moves at a constant velocity, the translation cannot be correctly measured by the IMU due to zero acceleration. Therefore, we propose a flexible marginalization scheme to properly manage the data in the sliding window (refer to Section III-D).

C. Sliding Window Formulation

Table I describes the mathematical notation used in the SLAM algorithm, listed in the order they appear in the text. With the AoAs and IMU measurements, we can fuse them to solve the SLAM problem. In this topic, there exists many sensor fusion methods, *e.g.*, EKF, particle filter. However, they usually requires a good initialization, which is very hard to obtain by AoAs due to the lack of metric scale information. Moreover, although filter-based approaches are very efficient in computation as they only estimate the current robot state and the map, the main drawback is that fixing the linearization points early may lead to suboptimal results. Therefore, we employ a graph-based SLAM framework in that 1) it achieves better performance via repetitively linearizing past robot states [23]; 2) it is insensitive to the initialization as the multi-view constraint helps recover the initial state.

TABLE I
MATHEMATICAL NOTATION.

Symbol	Description
$\hat{(\cdot)}$	the quantity that can be measured by sensors
$\boldsymbol{\mu}_i$	the hidden state at discrete timestamp i
$\mathbf{o}_i^j$	the AoA of backscatter j at timestamp i
$\mathbf{b}_j$	the position of backscatter j
$\mathbf{u}_{i+1}^i$	the translation between state $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_{i+1}$
$\mathcal{S}$	the state vector in the sliding window
$\mathcal{A}$	the set of AoA measurements in the window
$\mathcal{I}$	the set of IMU measurements in the window
$\mathbf{Q}_i^j$	the information matrix of the AoA constraint
Ω_i^j	the AoA covariance matrix
θ	the corrected AoA
$\mathbf{r}_i^j$	the direction vector referred to i^{th}tag at time j
d_i^j	the distance of the robot to i^{th}tag at time j
$\mathbf{n}_i^j$	the measurement noise
$\mathbf{P}_{k+1}^k$	the information matrix of odometry constraint
$\boldsymbol{\Lambda}_{k+1}^k$	the IMU covariance matrix
$\mathbf{a}_t$	the acceleration at current time t
$\boldsymbol{\omega}_t$	the angular rate at current time t
$\mathbf{R}_t^k$	the rotation matrix from time k to current t
$\mathbf{V}_{k+1}^k$	the velocity between time k and $k+1$
$\mathbf{T}_{k+1}^k$	the translation between time k and $k+1$
$\boldsymbol{\nu}_k$	the velocity at time k
$\mathbf{g}$	the earth's vertical gravity
Δt	the time interval between two measurements

Fig. 7 shows the graph representation of our SLAM formulation. Let $\boldsymbol{\mu}_i$ denote the hidden state at discrete timestamp i. At each timestamp, the robot observes a set of AoAs from multiple backscatters. $\mathbf{o}_i^j$ is the geometric observation from the AoA of backscatter j at timestamp i and $\mathbf{b}_j$ denotes the position of backscatter j. The relative translation between two robot states $\boldsymbol{\mu}_i$ and $\boldsymbol{\mu}_{i+1}$ is captured by an odometry edge $\mathbf{u}_{i+1}^i$, which can be obtained by IMU preintegration techniques [24].

We define the state vector in the sliding window that merges the hidden variables of robot and backscatter together,

$$\mathcal{S} = [\boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{n-1}, \mathbf{b}_0, \mathbf{b}_1, \ldots, \mathbf{b}_{m-1}]^\top, \tag{4}$$

where the initial position $\boldsymbol{\mu}_0 = [0, 0, 0]$. All these variables refer to the world frame, which is related to the real world where the gravity is vertical. n is the number of robot's state in the sliding window, m denotes the number of observed backscatter tags, and $\mathbf{b}_i$ is the position of tag i in the world frame. At this stage, we have constructed the graph from the AoA observations and the IMU odometry. Next step we seek to find the configuration of the positions of the robot and tags that best satisfies the constraints, *i.e.*, the edges of the graph.

Since our system only involves translations, parameters in

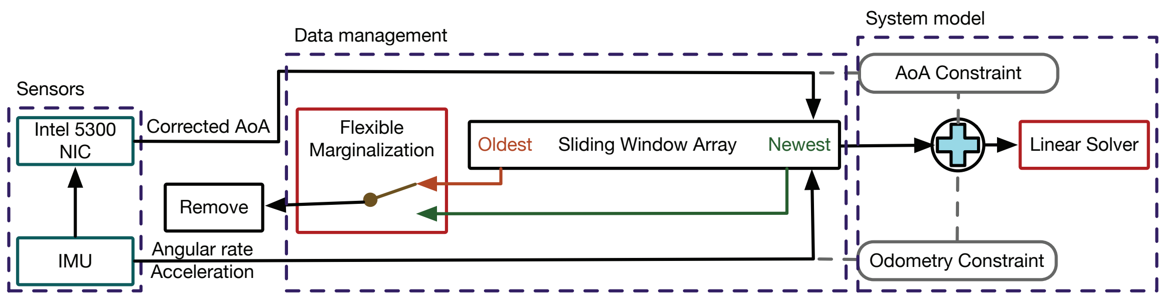

Fig. 6. The overview of the SLAM-based system.

$\mathcal{S}$ are in Euclidean space. We can formulate the problem as a linear problem and the optimal state sequence $\mathcal{S}^*$ in the sliding window can be estimated by solving:

$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}} \Big\{ \overbrace{\mathbf{A}(\mathcal{S})}^{\text{AoA constraint}} + \overbrace{\mathbf{D}(\mathcal{S})}^{\text{odometry constraint}} \Big\}, \tag{5}$$

where

$$\begin{aligned} \mathbf{A}(\mathcal{S}) &= \sum_{(i,j)\in\mathcal{A}} \left\| \hat{\mathbf{o}}_i^j - \mathbf{Q}_i^j \mathcal{S} \right\|^2_{\mathbf{\Omega}_i^j} \\ \mathbf{D}(\mathcal{S}) &= \sum_{k\in\mathcal{I}} \left\| \hat{\mathbf{u}}_{k+1}^k - \mathbf{P}_{k+1}^k \mathcal{S} \right\|^2_{\mathbf{\Lambda}_{k+1}^k} . \end{aligned} \tag{6}$$

$\mathcal{A}$ denotes the set of AoA measurements between all tags and the robot in the window. $\mathcal{I}$ denotes the set of all inertial measurements in the window. The constraints are the sum of the Mahalanobis norm of their measurement errors. Specifically, the AoA constraint is

$$\left\| \hat{\mathbf{o}}_i^j - \mathbf{Q}_i^j \mathcal{S} \right\|^2_{\mathbf{\Omega}_i^j} = \left(\hat{\mathbf{o}}_i^j - \mathbf{Q}_i^j \mathcal{S} \right)^\top \left(\mathbf{\Omega}_i^j \right)^{-1} \left(\hat{\mathbf{o}}_i^j - \mathbf{Q}_i^j \mathcal{S} \right), \tag{7}$$

and the odometry constraint is

$$\begin{aligned} &\left\| \hat{\mathbf{u}}_{k+1}^k - \mathbf{P}_{k+1}^k \mathcal{S} \right\|^2_{\mathbf{\Lambda}_{k+1}^k} = \\ &\left(\hat{\mathbf{u}}_{k+1}^k - \mathbf{P}_{k+1}^k \mathcal{S} \right)^\top \left(\mathbf{\Lambda}_{k+1}^k \right)^{-1} \left(\hat{\mathbf{u}}_{k+1}^k - \mathbf{P}_{k+1}^k \mathcal{S} \right). \end{aligned} \tag{8}$$

To solve this system, the terms of the AoA constraint $\left\{ \hat{\mathbf{o}}_i^j, \mathbf{Q}_i^j, \Omega_i^j \right\}$ and the odometry constraint $\left\{ \hat{\mathbf{u}}_{k+1}^k, \mathbf{P}_{k+1}^k, \mathbf{\Lambda}_{k+1}^k \right\}$ need to be defined.

AoA constraint. The direction vector $\mathbf{r}_i^j$ referred to the observed i^{th}tag at timestamp j can be defined by the AoA θ as $\mathbf{r}_i^j = [\cos(\theta), \sin(\theta), 0]^\top$. With an unknown distance d_i^j, a simple geometric relationship can be expressed as

$$d_i^j \mathbf{r}_i^j = \mathbf{R}_0^j \left(\mathbf{b}_i - \boldsymbol{\mu}_j \right), \tag{9}$$

where $\mathbf{b}_i$ is the i^{th}tag's position and $\boldsymbol{\mu}_j$ is the robot position at timestamp j. Since $\mathbf{r}_i^j$ should have the same direction as the vector $\mathbf{b}_i - \boldsymbol{\mu}_j$ if there is no measurement noise. The expected observation can be expressed by a cross product operation,

$$\hat{\mathbf{o}}_i^j = \hat{\mathbf{0}} = \left(\mathbf{R}_j^0 \mathbf{r}_i^j \right) \times \left(\mathbf{b}_i - \boldsymbol{\mu}_j \right) = \mathbf{Q}_i^j \mathcal{S} + \mathbf{n}_i^j, \tag{10}$$

where $\mathbf{n}_i^j$ denotes the noise, assuming that it follows a Gaussian distribution. The AoA covariance $\mathbf{\Omega}_i^j$ can be pre-measured by statistical methods and updated along the optimization process. Initially, the distance d_i^j is given by a reasonable guess. Then it will be refined automatically along the sliding window optimization as the positions of the robot and tags are updated. Therefore, the initial guess is insensitive in our system.

Note that we consider the AoA in 2D case for the ease of representation. Our system can be trivially extended to work in 3D case. The circular antenna array we use is capable of measuring azimuth θ angle and elevation angle ψ for 3D AoA representation. The direction vector becomes $\mathbf{r} = [\cos\theta \sin\psi, \ \sin\theta \sin\psi, \ \cos\psi]$. However, we can no longer employ the joint AoA-ToF estimation technique [20] to obtain the 3D AoA as the joint parameter searching process of this technique will increase the computation complexity exponentially due to the additional parameter, *i.e.*, ψ. To reduce the complexity, we employ an additional parameter search instead of the joint search. This is an approximate solution of [20] that slightly sacrifices the accuracy to significantly save the computation cost. Its computation complexity remains the same as the 2D case. It may occasionally miss the optimal parameter configuration but the overall performance is very close to the optimal solution as proved by [25]. Since the 3D extension is incremental to our contribution, we omit the details in this paper.

Odometry constraint. Typically, the data rate of IMU is higher than AoA rate. Given two consecutive timestamps $[k, k+1]$ at which the AoAs from multiple tags are received, there have been multiple buffered inertial measurements, which include acceleration $\mathbf{a}_t \in \mathbb{R}^3$ and angular rate $\boldsymbol{\omega}_t \in \mathbb{R}^3$. We can preintegrate them to obtain an overall odometry representation between $\boldsymbol{\mu}_k$ and $\boldsymbol{\mu}_{k+1}$ as follows:

$$\begin{aligned} \mathbf{V}_{k+1}^k &= \sum_{t\in[k,k+1]} \mathbf{R}_t^k \mathbf{a}_t \Delta t \\ \mathbf{T}_{k+1}^k &= \sum_{t\in[k,k+1]} \left[\mathbf{V}_{k+1}^k \Delta t + \mathbf{R}_t^k \mathbf{a}_t \Delta t^2 \right], \end{aligned} \tag{11}$$

where $\mathbf{R}_t^k = \sum_{i\in[k,t]} \left[\mathbf{R}_i^k \lfloor \omega_t \times \rfloor \Delta t \right]$, $\mathbf{R}_t^k \in \mathrm{SO}(3)$. $\lfloor \omega_t \times \rfloor$ is the skew-symmetric matrix from $\boldsymbol{\omega}_t$, Δt the time interval between two consecutive measurements. $\mathbf{R}_t^k$ denotes the incremental rotation from time k to current time t, which is available through short-term integration of gyroscope measurements. Then, we can write the propagation model of positions as

$$\boldsymbol{\mu}_{k+1} = \boldsymbol{\mu}_k + \mathbf{R}_k^0 \boldsymbol{\nu}_k \Delta t - \mathbf{R}_k^0 \mathbf{g} \Delta t^2/2 + \mathbf{R}_k^0 \mathbf{T}_{k+1}^k, \tag{12}$$

where $\mathbf{T}_{k+1}^k$ can be obtained by Eqn. (11). $\mathbf{g} = [0, 0, 9.8]^\top$

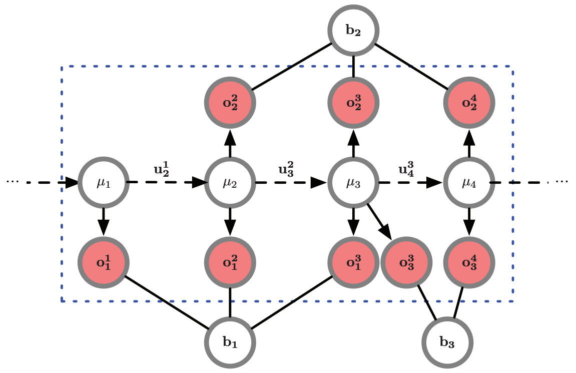

Fig. 7. Graph representation for sliding window AoA-SLAM. $\boldsymbol{\mu}$ is the hidden state of robot position; $\mathbf{b}$ denotes the hidden state of backscatter position; $\mathbf{o}$ denotes the AoA observation; $\mathbf{u}$ is the odometry captured by IMU. The sliding window represented by the blue dashed box contains four states and their observed AoAs.

is the vertical gravity. Since the robot only moves in a room (assuming a horizontal plane), it is safe to obtain the accelerations that account for motions by directly subtracting the gravity. $\boldsymbol{\nu}_k$ denotes the velocity at timestamp k. It can be propagated as

$$\boldsymbol{\nu}_{k+1} = \mathbf{R}_k^{k+1}\boldsymbol{\nu}_k - \mathbf{R}_k^{k+1}\mathbf{g}\Delta t + \mathbf{R}_k^{k+1}\mathbf{V}_{k+1}^k, \tag{13}$$

where $\mathbf{V}_{k+1}^k$ is obtained from Eqn. (11). $\mathbf{R}_k^0$ is the change in rotation since the initial state. We can see that the update equation for the quantity $\boldsymbol{\mu}_k$ and $\boldsymbol{\nu}_{k+1}$ will be linear in Eqn. (12) and Eqn. (13) if rotation $\mathbf{R}_k^0$ are provided. This rotation can be obtained by solving a linear system that incorporate the short-term integration of gyroscope measurements. For brevity, we omit the details and refer to the broad literature discussing these ideas [26].

Accordingly, Eqn. (12) can be rewritten as a linear function of the state $\mathcal{S}$:

$$\begin{aligned}\hat{\mathbf{u}}_{k+1}^k = \hat{\mathbf{T}}_{k+1}^k &= \mathbf{R}_0^k\left(\boldsymbol{\mu}_{k+1} - \boldsymbol{\mu}_k\right) - \boldsymbol{\nu}_k\Delta t + \mathbf{g}\frac{\Delta t^2}{2} \\ &= \mathbf{P}_{k+1}^k\mathcal{S} + \mathbf{n}_{k+1}^k,\end{aligned} \tag{14}$$

where $\boldsymbol{\nu}_k$ can be updated by Eqn. (13), $\mathbf{n}_{k+1}^k$ denotes the additive measurement noise. Typically, we assume the additive noise follows a Gaussian distribution. Then the covariance $\boldsymbol{\Lambda}_{k+1}^k$ can be calculated using the pre-integration technique proposed in [27].

At this point, all constraints in Eqn. (6) are explicitly defined. The information matrices and state vectors in the sliding window can be stacked to construct a large array of linear equations so that the positions of the robot and the tags in the window can be solved altogether.

However, the robot may undergo some degenerated motions in practical, causing the data marginalization problem as mentioned in Section III-B. We next elaborate on our novel marginalization scheme.

D. Flexible Marginalization

The FIFO marginalization scheme works fine when the robot is moving with non-zero acceleration. However, this scheme fails when the robot performs degenerate motions (zero acceleration), *e.g.*, being stationary or moving in a constant velocity. In these cases, the IMU odometry, *i.e.*, the translations between AoAs, cannot be correctly measured, failing to recover the metric scale. Unfortunately, zero acceleration motion is unavoidable in practice for a mobile robot and it must be handled properly.

When being stationary, the FIFO scheme results in that all measurements in the sliding window come from the same position. The translation between two AoAs is unobservable so that we cannot recover the metric scale. Intuitively, the last-in-first-out (LIFO) sliding window scheme can preserve the scale observability. In this case, we only update the position of the robot because LIFO scheme does not keep new AoAs.

When moving at a constant velocity, the translations between AoAs cannot be correctly measured, making the metric scale still unobservable. For example, if the robot first undergoes generic motions with sufficient accelerations $(\boldsymbol{\mu}_0, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{l-1})$ and then enters a constant velocity motion $(\boldsymbol{\mu}_l, \boldsymbol{\mu}_{l+1}, \ldots, \boldsymbol{\mu}_{l+n-1})$, the scale can only be observed when the states correspond to generic motions are included in the sliding window. However, this will inevitably increase the computation complexity so that the limited computation source of the robot cannot ensure the real-time property. A promising solution is to provide an initial estimate of $\boldsymbol{\mu}_l$, then we can propagate the scale from $\boldsymbol{\mu}_{l-1}$ to $\boldsymbol{\mu}_l$. This can be done by proper marginalization of $\boldsymbol{\mu}_{l-1}$ as it is removed from the sliding window at step $l+n$.

Based on the above discussion, we propose a flexible marginalization scheme to address the issue of degenerated motions. Consider a full state vector $\mathcal{S} = [\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_{n-1}|b_{\mathcal{L}}]$ where $b_{\mathcal{L}}$ denotes the set of all observed backscatters in the sliding window. We add a state with a new AoA observation $\boldsymbol{\mu}_n$ to the sliding window if any of the following three criteria are satisfied:

- The time between two AoAs Δt is larger than δ.
- The observed backscatter tags change in the new state.
- The newest AoA observation significantly differs from the second newest observation in the sliding window.

The first criterion aims to bound the error in the integrated result of IMU measurements between two AoAs. Through some tests, we empirically set δ to be 500 ms. The second criterion indicates that the system observes new tags that are needed to be localized. The third criterion aims to ensure that the translation of the robot with respect to the observed AoAs is significant.

To quantify the difference of AoA observations, we define the *similarity* between two AoA observations. At each timestamp, an AoA observation is a set of AoAs from multiple backscatters. For the AoA θ_i^j of $i$thbackscatter observed at timestamp j, we have its direction vector $\mathbf{r}_i^j = [\cos(\theta_i^j), \sin(\theta_i^j)]^\top$. Then we define the AoA observation at timestamp j as

$$\mathbf{O}^j = [\mathbf{r}_1^j, \mathbf{r}_2^j, \ldots, \mathbf{r}_m^j], \quad \mathbf{O}^j \in \mathbb{R}^{2\times m}, \tag{15}$$

where m is the number of observed backscatters at time j. For any timestamp $k > j$ that the observed backscatters remain

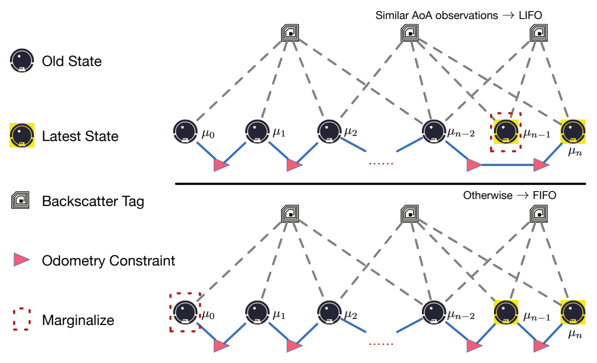

Fig. 8. An illustration of the flexible marginalization. If the second latest state has a similar AoA observation to the latest one, we will simply marginalize it and all its corresponding AoA measurements. However, pre-integrated odometry measurements are kept and the pre-integration process is continued towards the next state. Otherwise, we will keep it in the window and marginalize the oldest state and its corresponding AoA and odometry measurements. The information of marginalized states is turned into a prior.

unchanged, the similarity can be defined as

$$\mathcal{M}_{jk} = 1 - \frac{1}{m}\sum_{i=1}^{m}\left(\mathbf{O}^j(i)^\top \cdot \mathbf{O}^k(i)\right), \tag{16}$$

where $\mathbf{O}^j(i)$ denotes i^{th}column of $\mathbf{O}^j$. The similarity $\mathcal{M} \in [0, 2]$. The smaller $\mathcal{M}$ the more similar AoA observations. Through experiments, we empirically set a threshold ε. When the similarity of the most recent two AoA observations is larger than ε, it satisfies the third criterion.

The pseudo code is shown in Algorithm 1. The algorithm requires that all newly added AoAs $b_{\mathcal{L}^+}$ to have at least two observations to succeed in recovering the scale via triangulation (Line 1). Then we set a variable f = LIFO/FIFO to indicate whether the system marginalizes out the second newest state $\boldsymbol{\mu}_{n-1}$ or the oldest one $\boldsymbol{\mu}_0$. The value of f is determined based on whether the new state observed a new tag or the similarity $\mathcal{M}^n_{n-1}$ between two most recent AoA observations (Lines 5–9 and Lines 14–18).

To marginalize a chosen state, we first construct a new prior based on all measurements related to the removed state (Lines 3 and 11). We then remove the state, the corresponding AoA observation, and the backscatter tags $b_{\mathcal{L}^-}$ that are first observed by it (Lines 4 and 12). The new prior can be expressed as

$$\mathbf{\Gamma}_p^+ = \mathbf{\Gamma}_p + \sum_{(i,j)\in\mathcal{A}^-}\left(\mathbf{Q}_i^j\right)^\top\left(\mathbf{\Omega}_i^j\right)^{-1}\mathbf{Q}_i^j + \sum_{k\in\mathcal{I}^-}\left(\mathbf{P}_{k+1}^k\right)^\top\left(\mathbf{\Lambda}_{k+1}^k\right)^{-1}\mathbf{P}_{k+1}^k, \tag{17}$$

where $\mathcal{A}^-$ and $\mathcal{I}^-$ are the sets of removed AoA and IMU measurements respectively. The marginalization can be carried out via Schur Complement [26]. The prior $\mathbf{\Gamma}_p$ is the initial condition computed by solving system (5). Eqn. (17) converts the sum of the Mahalanobis norm corresponding to the removed measurements into a new prior. Note that in the LIFO scheme we have an additional operation that concatenates the IMU odometry from μ_{n-2} to μ_n for preserving additional motional information (Line 13).

This approach tries to preserve all information provided by the marginalized states. On one hand, our approach keeps removing the most recent state if the robot has small motion or is stationary. Keeping older states in this case can preserve the non-zero acceleration information that helps recover the scale. On the other hand, when the robot undergoes a constant velocity motion, older states will be removed and the priors implicitly propagate the scale information forward for subsequent estimates. Fig. 8 illustrates the two working cases of the flexible marginalization approach. In this way, our system needs to incorporate the prior information as follows:

$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmin}}\left\{\overbrace{\left(\mathbf{b}_p - \mathbf{\Gamma}_p\mathcal{S}\right)}^{\text{Prior}} + \mathbf{A}(\mathcal{S}) + \mathbf{D}(\mathcal{S})\right\}, \tag{18}$$

where $\left\{\mathbf{b}_p, \mathbf{\Gamma}_p\right\}$ is the prior for our system. The system is then solved with all available measurements within the sliding window plus any available prior (Line 20).

Algorithm 1 Flexible Marginalization

```
Require:
    $\mathcal{S} \leftarrow [\boldsymbol{\mu}_0, \ldots, \boldsymbol{\mu}_{n-1} | b_{\mathcal{L}}]$
    $f$ = FIFO or LIFO
    $\{\mathbf{b}_p, \mathbf{\Gamma}_p\} \leftarrow$ Prior
Ensure: $\Delta t > \delta$ or $b_{\mathcal{L}}$ changes or $\mathcal{M}^n_{n-1} > \varepsilon$
1: $\mathcal{S} \leftarrow \mathcal{S} \cup [\boldsymbol{\mu}_n | b_{\mathcal{L}^+}]$
2: if $f$ = FIFO then
3:     $\{\mathbf{b}_p, \mathbf{\Gamma}_p, b_{\mathcal{L}^-}\} \leftarrow$ Marginalization($\boldsymbol{\mu}_0$)
4:     $\mathcal{S} \leftarrow \mathcal{S} \backslash [\boldsymbol{\mu}_0 | b_{\mathcal{L}^-}]$
5:     if $\mathcal{M}^n_{n-1} > \varepsilon$ or $b_{\mathcal{L}}$ changes then
6:         $f \leftarrow$ FIFO
7:     else
8:         $f \leftarrow$ LIFO
9:     end if
10: else
11:     $\{\mathbf{b}_p, \mathbf{\Gamma}_p, b_{\mathcal{L}^-}\} \leftarrow$ Marginalization($\boldsymbol{\mu}_{n-1}$)
12:     $\mathcal{S} \leftarrow \mathcal{S} \backslash [\boldsymbol{\mu}_{n-1} | b_{\mathcal{L}^-}]$
13:     OdometryConcatenation($\boldsymbol{\mu}_{n-2}, \boldsymbol{\mu}_n$)
14:     if $\mathcal{M}^n_{n-2} > \varepsilon$ or $b_{\mathcal{L}}$ changes then
15:         $f \leftarrow$ FIFO
16:     else
17:         $f \leftarrow$ LIFO
18:     end if
19: end if
20: Solve $\mathcal{S}$ using (18) and (6) with $\{\mathbf{b}_p, \mathbf{\Gamma}_p\}$
21: return $\{f, \mathbf{b}_p, \mathbf{\Gamma}_p, \mathcal{S}\}$
```

IV. Implementation and Evaluation

A. Implementation and Experimental Setup

We implemented Rover on an Intel NUC with a 1.3 GHz Core i5 processor with 4 cores, an 8 GB of RAM and a 120 GB SSD, running Ubuntu Linux equipped with Intel 5300 NICs and a LORD MicroStrain 3DM-GX4-45 IMU. We use

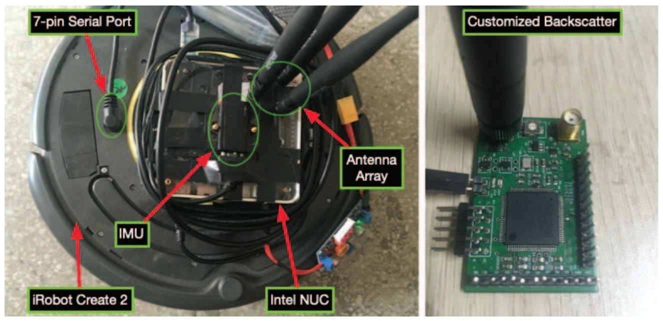

Fig. 9. The experimental platform. The left shows the receiver attaches on the robot and sends commands to control its motions through the Create's 7-pin serial port. The right shows one of our customized backscatter tags.

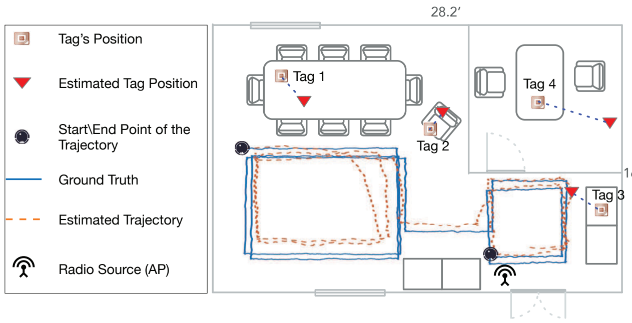

Fig. 10. We use the NUC to control a robot moves in a pre-defined rectangular trajectory in the meeting room. The ground truth is provided by the program of defining the trajectory that runs in the NUC.

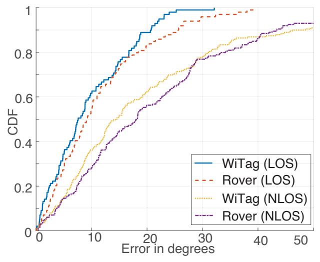

Fig. 11. The accuracy of AoA estimation.

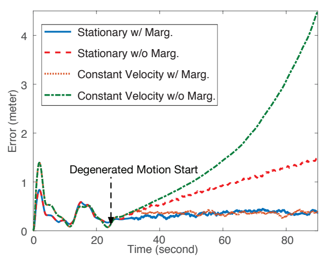

Fig. 12. The performance of robot localization under degenerated motions.

TABLE II
COMPUTATION TIME FOR EACH UPDATE.

No. of States	Robot Position Accuracy (cm)	Mean Computation Time (ms)
20	70.2	18.29
30	45.3	27.40
40	38.6	39.21
50	36.5	58.50
60	37.2	99.38
70	37.0	158.17
80	35.9	235.42

the Linux 802.11 CSI tool [17] to obtain the wireless channel information for each packet. Thanks to the open-source hardware of HitchHike [18], we build the customized tags to backscatter commodity WiFi signals. The power consumption of the tags is only 33μW, 1000$\times$ lower than the mW-level power consumption of commodity WiFi. The whole system is implemented in C++. The NUC connects to the iRobot Create 2 and uses ROS (Robot Operating System) as the interfacing robotics middleware to control the robot's moving trajectory. The experimental platform is shown in Fig. 9.

The experiments are conducted in a 9×5 square meters meeting room in our laboratory, which is a typical indoor setting. Four backscatter tags are deployed in the room. Each tag is configured to shift a frequency and backscatter signals in a separate channel. This prevents the interference between tags. In addition, the frequency shift can be an identifier to distinguish the received signal from which tag as each tag occupies a separate channel.

B. Micro-benchmark Evaluation

Backscatter AoA Estimation. We first test the accuracy of tag-to-receiver AoA estimation. The key difference in AoA estimation from the state-of-the-art [12], WiTag, is that we empower it with time-division multiplexing so that the receiver can simultaneously measure AoAs of multiple tags who backscatter signals in different channels. We demonstrate the AoA estimation by four tags deployed in line-of-sight (LOS) and non-LOS (NLOS) settings. The CDF plotted in Fig. 11 shows that the performance of Rover is similar to WiTag. The median errors of Rover and WiTag are 9.3° and 8.1° respectively in LOS deployment. In NLOS deployment, the median errors of Rover and WiTag are 18.1° and 14.6°, respectively.

Performance under degenerated motions. We then test Rover's tracking performance under degenerated motions, including being stationary and moving at a constant velocity. Meanwhile, we run Rover in two concurrent processes, with and without marginalization, for the evaluation of our marginalization algorithm. The robot first undergoes generic motions in both tests. Then at 24 second, it stays stationary in the first test and moves at a constant velocity of 0.1 m/s in the second test. Fig. 12 shows that the localization error accumulates in both cases of degenerated motions if there is no marginalization, but exhibits no accumulation when applied. The robot's mean localization errors are 35.5 cm and 33.7 cm during the first 24 seconds of generic motions. Then the errors go up to 82.1 cm and 148.7 cm when being stationary and moving at a constant velocity in the case of no marginalization. When applying marginalization, the errors reduce to 39.1 cm and 37.0 cm in the two tests. We notice that the error accumulates faster in the constant velocity movement when marginalization is absent. This is due to the fact that near-zero linear acceleration in this case makes the moving distance unobservable from IMU measurements. Meanwhile, the AoA estimation still changes according to the movements, yielding erroneous results in the SLAM framework. In contrast, when being stationary, the measurements from the AoA and IMU do not contradict each other.

Fig. 13 depicts the localization error of a backscatter tag in LOS deployment in different degenerated motions, with and without marginalization. The mean errors of the two tests under generic motions are 76.5 cm and 83.4 cm. When the

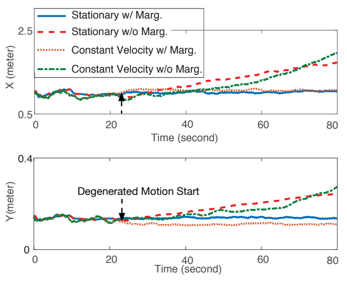

Fig. 13. The performance of tag localization under degenerated motions.

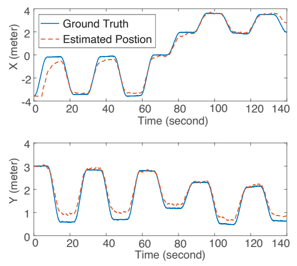

Fig. 14. Robot position tracking in the meeting room.

marginalization is not applied, the final errors rise to 118.7 cm in stationariness and 121.0 cm in constant velocity motion, respectively. In contrast, applying our marginalization algorithm eliminates the error accumulation. The errors remain 71.3 cm and 79.6 cm respectively, which are similar to the performance under generic motions. Again, the error in constant velocity motion accumulates faster due to the erroneous computation.

Rover's complexity. The real-time processing is a desired property in that the real-time location estimates can be used for navigating the robot. Thus, the computation complexity analysis is required. The most time-consuming part of Rover is the SLAM framework, which is a linear system that can be computed quite efficiently. Specifically, we use the standard Cholesky decomposition implemented by Eigen to solve the linear system. The time complexity is $O(N^3)$ in theory, where N denotes the number of states. In practice, the multithreaded routines make the computation time be approximate N^2 growth. Despite the mild time complexity, we further employs a sliding window formulation to ensure the real-time processing by bounding the parameter N. This is because N, which means the number of states in Rover, can be vast in a long-term run and thus significantly increases the computational cost if we solve the full batch SLAM for the best possible accuracy. It poses a tradeoff between localization accuracy and computation time. Essentially, the more states involved the more accurate results obtained. But this inevitably results in higher delay since a larger state vector and the corresponding measurements are involved in the optimization framework. To shed light on that, we tune the number of states in the sliding window from 20 to 80 to seek a balance between accuracy and computational cost. Table II lists the results in different amounts of states considered. When incorporating more than 50 states, we can see a marginal increase of the accuracy and a significant increase of the computation time, which goes up to hundreds of milliseconds. Therefore, in our experiments, we empirically set the size of sliding window to be 50. The overall average computation time is 58.50 ms for each update and thus Rover achieves the real-time processing.

C. System-level Evaluation

Generic motions. Fig. 10 shows the system deployment and the overall performance of Rover. The performance of tracking the robot's trajectory is plotted in Fig. 14. The mean error over the estimated trajectory is 39.3 cm. The accuracy goes beyond the expectation from the noisy AoA estimation (Fig. 11). This is because our system introduces the inertial sensors that provide an additional sensing modality for positioning. Moreover, our sliding window optimization filters out the noise of heterogeneous sensors by finding the configuration of positions that best fits different spatial measurement constraints.

Fig. 15 shows the localization results of the tags. Initially, the tags' locations are set to be $(0,0)$. After about 20 seconds, Rover localizes tags 1 and 2 as their AoAs are available. Tags 3 and 4 are localized at about 80 and 90 seconds later as the robot approaches them and received their backscattered packets. Meanwhile, Rover stops updating the location of tag 1 after about 95 seconds as it loses contact with the robot. The four tags' final localization errors are 73.6 cm, 52.9 cm, 97.2 cm, and 145.9 cm, respectively. Among them, the error of tag 4 is higher due to its NLOS deployment. The mean localization error in LOS deployment is 74.6 cm.

Mixed with degenerated motions. To highlight the effectiveness of our marginalization approach in coping with degenerated motions, we control the robot to stop for a while at some points in the original trajectory and concurrently run Rover in two processes, *i.e.*, with and without the marginalization respectively. Fig. 16 shows the position tracking results. The robot stops at 36thsecond and 125thsecond, both for a period of 30 seconds. When incorporating marginalization, there is no sign of error accumulation during the stationary periods. The mean error over the whole trajectory is 43.2 cm. In contrast, without marginalization, the mean error rises to 83.4 cm.

Fig. 17 depicts the localization performance of four tags. For a clearer demonstration, we zoomed into the degenerated periods of each tag. For tags 1 and 2, they experience two periods of being stationary at 36thsecond and 125thsecond. In the absence of the marginalization, the two tags' final localization errors increase to 146.8 cm and 182.3 cm. On the other hand, when enabling the marginalization of Rover, the final errors remain almost unchanged that they are still at 46.4 cm and 79.3 cm for tags 1 and 2. After about 155 seconds, tag 1's location is no longer updated as it loses the contact with the robot. Similar situation appears on tags 3 and 4. Their final errors rise to 124.3 cm and 178.9 cm without the marginalization, and remain at 78.2 cm and 129.2 cm when enabling the marginalization, which are similar to the performance under generic motions

In summary, the localization accuracy is decimeter-level, which is similar to the state-of-art WiFi based localization systems [12], [20]. The uniqueness of Rover is that it works without landmarks or any map of the environment, while conventional solutions need multiple APs with known positions. Conventional solutions use more APs to provide redundant positioning measurements and combat the noise of WiFi measurements. On the contrary, we take advantage of IMU and a robot's mobility to enable a new localization paradigm. The inertial measurements play the role of combating the WiFi noise and the drift-free localizability of WiFi helps correct the IMU drift in return. To bound the computation complexity, we employ a sliding window based formulation and incur a marginalization issue under degenerated motions. Our flexible marginalization algorithm succeeds in addressing the issue.

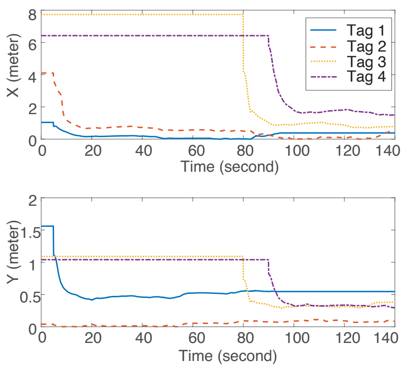

Fig. 15. The localization errors of four tags in the meeting room.

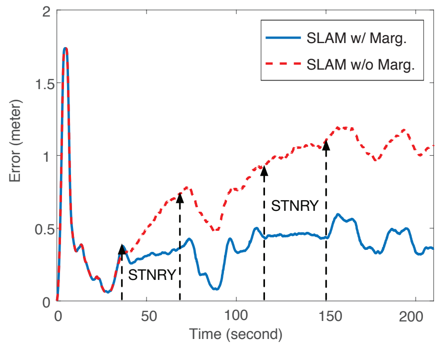

Fig. 16. Robot position tracking with degenerated motions. **STNRY** represents **stationary**.

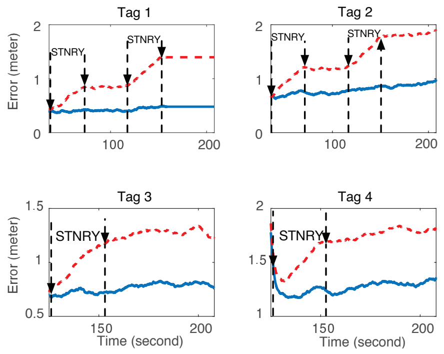

Fig. 17. The error reports of four tags' positioning. They are zoomed in to the stationary periods for a better comparison. **STNRY** represents **stationary**.

V. Related Work

Backscatter technology. Backscatter communication technologies have attracted significant attentions in the last few years to enable low-power and long-term communications [2]–[8], [10], [18], [28]–[30]. Hessar *et al.* [8] proposes a wireless protocol for backscatter networks that supports hundreds of concurrent transmissions. These technologies bring the vision of ubiquitous connectivity into reality for the next-generation of IoT.

IoT localization. For many IoT applications, finding such IoT devices is crucial for their smart services. Battery-free RFIDs have been used for localizing IoT devices. Wang *et al.* [31] exploits multipath to accurately locate RFIDs in indoors. Furthermore, Luo *et al.* [11] stitches multiple packets to expand the signal bandwidth and improve the ranging accuracy for 3D RFID tracking. But these approaches need to deploy dedicated RFID readers and they suffers from a limited communication range. To address these issues, Kotaru *et al.* [12] presents a WiFi backscatter based localization system. It analyzes the phase information when the received WiFi packet traverses two links, the AP-to-tag link and the tag-to-receiver link. Then it measures the AoAs of a tag to receivers for localization. Despite the high accuracy, all these approaches require to deploy multiple landmarks, *e.g.*, WiFi APs and RFID readers, whose locations are known to enable the localizability. Calibrating these landmarks to obtain their locations is usually labor-intensive and thus has high start-up cost. In contrast, Rover is plug-and-play to localize IoT devices with backscatter tags without any landmarks or environment knowledge.

Visual/laser SLAM. The core of Rover is a SLAM framework that simultaneously localizes the robot and the connected tags via the backscatter RF sensing modality. SLAM has been a long studied problem in the robotic community. Monocular visual SLAM approaches [32], [33] leverage visual data from a monocular camera to localize the robot who equips with the camera and the map represented by point cloud. Shen *et al.* [23] fuses monocular visual data with IMU measurements to estimate the metric scale of the environment. But the visual sensing is very sensitive to lighting conditions and environmental texture. In addition, laser based SLAM frameworks use laser ranging sensors, *e.g.*, LiDAR, to obtain 3D structure of environments [34], [35]. The sensors use emitted light so that they works independent of the ambient light. However, the laser pulse is sensitive to LOS interference like fog or smoke. Overall, visual and laser based SLAM approaches use light-based sensing modalities to estimate the depth map of environments, which are highly accurate and reliable but limited by lighting and LOS conditions. In contrast, Rover's SLAM takes backscatter RF signals as a bridge to connect the robot with the surrounding environment. The RF sensing is complementary to the visual/laser sensing as RF signals can propagate in NLOS settings, traversing obstacles like walls and furnitures thanks to their larger wavelength. The environment in our work is represented by the RF map, *i.e.*, the locations of backscatters. Thus, we localize the tags when solving the mapping problem in our SLAM approach.

RF-based SLAM. There have been many SLAM systems that work with RF signals, *e.g.*, WiFi and UWB [36]–[42]. B. Ferris *et al.* [43] and Huang *et al.* [37] take the received signal strength of WiFi signal to do the SLAM in indoor environments. They use Gaussian Process Latent Variable Model to reduce the high-dimensional fingerprints to latent-space locations. The RSSI is prone to be noisy in indoors so that the accuracies of these approaches are limited. Recently, Venkatnarayan *et al.* [41] leverages the drift-free localizability of WiFi to correct the drift of IMU and localize users. It can accurately track users' trajectories but does not address the mapping problem, *i.e.*, localizing the WiFi APs. Li *et al.* [42] use the phase difference of WiFi signals from active radios to infer AoAs for localization. But it only works in outdoors without multipath fading. Gentner *et al.* [38] and Li *et al.* [40] exploit multipath components in indoor venues to simultaneously localize the user and the multipath reflection points. Their studies provide great inspirations for our work. Compare with them, the fundamental differences of Rover are that 1) we only use a small antenna array to estimate AoAs with low-power backscattered signals for localization; 2) we handle multipath fading in indoors without any assumptions of the multipath layout; 3) we leverage IMU to measure the metric scale of environments and employ a sliding window fashion to formulate the graph-based optimization problem for better localizing backscatter tags with bounded computation complexity. Moreover, our formulation is very insensitive to the initialization point, which is crucial for conventional filter-based approaches, *e.g.*, extended Kalman filter (EFK) [38], [40] and we address the marginalization problem under degenerated motions to make Rover more practical.

VI. Conclusion

We presented Rover, a backscatter localization system with an AoA-IMU SLAM framework. We formulated a sliding window based model that fused inertial measurements with the AoAs of backscatter tags to a robot measured by commodity WiFi to simultaneously estimate the locations of the robot as well as the connected tags. In addition, we addressed the practical issues of Rover, including real-time processing and data marginalization in degenerated motions. We implemented Rover on the iRobot Create 2 platform attached with an Intel NUC and an IMU. The experiments in both LOS and NLOS indoor settings showed that Rover achieves localization accuracy of tens of centimeters for both the robot and the backscatter tags without any prior knowledge of the work space. Extending our system to work with other wireless devices, such as iBeacon, for better accuracy is an important task for future work.

References

[1] S. Zhang, W. Wang, S. Tang, S. Jin, and T. Jiang, "Localizing backscatters by a single robot with zero start-up cost," in *Proc. IEEE GLOBECOM*, 2019.

[2] B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall, "Wi-Fi backscatter: Internet connectivity for RF-powered devices," in *ACM SIGCOMM COMP. COM.*, vol. 44, no. 4, 2014, pp. 607–618.

[3] Y. Peng, L. Shangguan, Y. Hu, Y. Qian, X. Lin, X. Chen, D. Fang, and K. Jamieson, "PLoRa: a passive long-range data network from ambient LoRa transmissions," in *Proc. ACM SIGCOMM*, 2018.

[4] C. Xu, L. Yang, and P. Zhang, "Practical backscatter communication systems for battery-free internet of things: A tutorial and survey of recent research," *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 16–27, 2018.

[5] D. Bharadia, K. R. Joshi, M. Kotaru, and S. Katti, "Backfi: High throughput wifi backscatter," in *Proc. ACM SIGCOMM*, 2015.

[6] F. Amato, H. M. Torun, and G. D. Durgin, "Rfid backscattering in long-range scenarios," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2718–2725, 2018.

[7] J. Guo, X. Zhou, S. Durrani, and H. Yanikomeroglu, "Design of non-orthogonal multiple access enhanced backscatter communication," *IEEE Trans. Wireless Commun.*, vol. 17, no. 10, pp. 6837–6852, 2018.

[8] M. Hessar, A. Najafi, and S. Gollakota, "Netscatter: Enabling large-scale backscatter networks," in *Proc. USENIX NSDI*, 2019.

[9] X. Xia, Y. Zheng, and T. Gu, "Ftrack: Parallel decoding for lora transmissions," in *Proc. ACM SenSys*, 2019.

[10] W. Liu, Y.-C. Liang, Y. Li, and B. Vucetic, "Backscatter multiplicative multiple-access systems: Fundamental limits and practical design," *IEEE Trans. Wireless Commun.*, vol. 17, no. 9, pp. 5713–5728, 2018.

[11] Z. Luo, Q. Zhang, Y. Ma, M. Singh, and F. Adib, "3d backscatter localization for fine-grained robotics," in *Proc. USENIX NSDI*, 2019.

[12] M. Kotaru, P. Zhang, and S. Katti, "Localizing low-power backscatter tags using commodity wifi," in *Proc. ACM CoNEXT*, 2017.

[13] L. Chen, J. Xiong, X. Chen, S. I. Lee, K. Chen, D. Han, D. Fang, Z. Tang, and Z. Wang, "Widesee: towards wide-area contactless wireless sensing," in *Proc. ACM SenSys*, 2019, pp. 258–270.

[14] Y. Ma, N. Selby, and F. Adib, "Drone relays for battery-free networks," in *Proc. ACM SIGCOMM*, 2017.

[15] X. Wang, L. Gao, S. Mao, and S. Pandey, "Csi-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, 2017.

[16] Y. He, J. Liang, and Y. Liu, "Pervasive floorplan generation based on only inertial sensing: feasibility, design, and implementation," *IEEE JSAC*, vol. 35, no. 5, pp. 1132–1140, 2017.

[17] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11n traces with channel state information," *ACM SIGCOMM Comput. Commun. Rev.*, 2011.

[18] P. Zhang, D. Bharadia, K. Joshi, and S. Katti, "Hitchhike: Practical backscatter using commodity wifi," in *Proc. ACM SenSys*, 2016.

[19] D. Vasisht, S. Kumar, and D. Katabi, "Decimeter-level localization with a single wifi access point," in *Proc. USENIX NSDI*, 2016.

[20] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *Proc. ACM SIGCOMM*, 2015.

[21] A. M. Sabatini, "Quaternion-based extended kalman filter for determining orientation by inertial and magnetic sensing," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 7, pp. 1346–1356, 2006.

[22] X. Lu, J. Liu, and H. Zhao, "Collaborative target tracking of iot heterogeneous nodes," *Measurement*, vol. 147, p. 106872, 2019.

[23] Y. Lin, F. Gao, T. Qin, W. Gao, T. Liu, W. Wu, Z. Yang, and S. Shen, "Autonomous aerial navigation using monocular visual-inertial fusion," *J. Field Robot.*, vol. 35, no. 1, pp. 23–51, 2018.

[24] D. F. S. D. Forster C, Carlone L, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation," in *Proc. Robot. Sci. Syst.*, 2015.

[25] Y. Xie, J. Xiong, M. Li, and K. Jamieson, "md-track: Leveraging multi-dimensionality in passive indoor wi-fi tracking," *Proc. ACM MobiCom*, 2019.

[26] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Initialization-free monocular visual-inertial state estimation with application to autonomous mavs," in *Experimental robotics*. Springer, 2016, pp. 211–227.

[27] T. Lupton and S. Sukkarieh, "Visual-inertial-aided navigation for high-dynamic motion in built environments without initial conditions," *IEEE Trans. Robot.*, vol. 28, no. 1, pp. 61–76, 2012.

[28] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: wireless communication out of thin air," in *ACM SIGCOMM COMP. COM.*, vol. 43, no. 4, 2013, pp. 39–50.

[29] P. Zhang, M. Rostami, P. Hu, and D. Ganesan, "Enabling practical backscatter communication for on-body sensors," in *Proc. ACM SIGCOMM*, 2016.

[30] B. Kellogg, V. Talla, S. Gollakota, and J. R. Smith, "Passive Wi-Fi: bringing low power to Wi-Fi transmissions," in *Proc. USENIX NSDI*, 2016.

[31] J. Wang and D. Katabi, "Dude, where's my card?: Rfid positioning that works with multipath and non-line of sight," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 51–62.

[32] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[33] A. S. Huang, A. Bachrach, P. Henry, M. Krainin, D. Maturana, D. Fox, and N. Roy, "Visual odometry and mapping for autonomous flight using an RGB-D camera," in *Robotics Research*. Springer, 2017, pp. 235–252.

[34] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot SLAM system for 3D LiDARs," in *Proc. IEEE IROS*, 2017.

[35] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LiDAR SLAM," in *Proc. IEEE ICRA*, 2016.

[36] L. Taponecco, A. A. D'Amico, and U. Mengali, "Joint toa and aoa estimation for uwb localization applications," *IEEE Trans. Wireless Commun.*, vol. 10, no. 7, pp. 2207–2217, 2011.

[37] J. Huang, D. Millman, M. Quigley, D. Stavens, S. Thrun, and A. Aggarwal, "Efficient, generalized indoor WiFi GraphSLAM," in *Proc. IEEE ICRA*, 2011.

[38] C. Gentner, T. Jost, W. Wang, S. Zhang, A. Dammann, and U.-C. Fiebig, "Multipath assisted positioning with simultaneous localization and mapping," *IEEE Trans. Wireless Commun.*, vol. 15, no. 9, pp. 6104–6117, 2016.

[39] Y. Wang and K. Ho, "Unified near-field and far-field localization for aoa and hybrid aoa-tdoa positionings," *IEEE Trans. Wireless Commun.*, vol. 17, no. 2, pp. 1242–1254, 2017.

[40] X. Li, E. Leitinger, M. Oskarsson, K. Åström, and F. Tufvesson, "Massive mimo-based localization and mapping exploiting phase information of multipath components," *IEEE Trans. Wireless Commun.*, vol. 18, no. 9, pp. 4254–4267, 2019.

[41] R. H. Venkatnarayan and M. Shahzad, "Enhancing indoor inertial odometry with wifi," *Proc. ACM UbiComp*, vol. 3, no. 2, p. 47, 2019.

[42] B. Li, S. Zhang, and S. Shen, "CSI-based WiFi-inertial state estimation," in *Proc. IEEE MFI*, 2016.

[43] B. Ferris, D. Fox, and N. D. Lawrence, "Wifi-slam using gaussian process latent variable models," in *Proc. IJCAI*, 2007.