
# TrackIO: Tracking First Responders Inside-Out

**Ashutosh Dhekne,** ***University of Illinois at Urbana-Champaign;***
**Ayon Chakraborty, Karthikeyan Sundaresan,**
**and Sampath Rangarajan,** ***NEC Labs America, Inc.***

https://www.usenix.org/conference/nsdi19/presentation/dhekne

**This paper is included in the Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI '19).**

**February 26–28, 2019 • Boston, MA, USA**

ISBN 978-1-931971-49-2

**Open access to the Proceedings of the 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI '19) is sponsored by**

NetApp®

# TrackIO: Tracking First Responders Inside-Out

Ashutosh Dhekne[†] * Ayon Chakraborty[⋆] Karthikeyan Sundaresan[⋆] Sampath Rangarajan[⋆]
[†]*University of Illinois at Urbana-Champaign,* [⋆]*NEC Labs America, Inc.*

## Abstract

First responders, a critical lifeline of any society, often find themselves in precarious situations. The ability to track them real-time in unknown indoor environments, would significantly contributes to the success of their mission as well as their safety. In this work, we present the design, implementation and evaluation of TrackIO—a system capable of accurately localizing and tracking mobile responders real-time in large indoor environments. TrackIO leverages the mobile virtual infrastructure offered by unmanned aerial vehicles (UAVs), coupled with the balanced penetration-accuracy tradeoff offered by ultra-wideband (UWB), to accomplish this objective directly from outside, without relying on access to any indoor infrastructure. Towards a practical system, TrackIO incorporates four novel mechanisms in its design that address key challenges to enable tracking responders (i) who are mobile with potentially non-uniform velocities (e.g. during turns), (ii) deep indoors with challenged reachability, (iii) in real-time even for a large network, and (iv) with high accuracy even when impacted by UAV's position error. TrackIO's real-world performance reveals that it can track static nodes with a median accuracy of about 1–1.5 m and mobile (even running) nodes with a median accuracy of 2–2.5 m in large buildings in real-time.

## 1 Introduction

**Tracking first responders:** First responders are integral to the safety and security of any community and to the society at large. However, they often find themselves in precarious and unknown environments, which poses a threat to their own safety (e.g. "entrapment" [16] faced by fire-fighters). Being able to accurately track first responders in indoor environments, allows a commander outside to better visualize and direct his responders appropriately. This not only helps address the situation efficiently but also ensures safety of the responders themselves—the latter can now view and track their own location with respect to the rest of the team.

**Applicability of current solutions:** The topic of indoor localization has seen many solutions in the past decade [4,9,53]. These can be broadly categorized under those that rely on indoor infrastructure (e.g. multitude of access points, RF/acoustic/infrared beacons, etc.) and those that do not (e.g. leveraging cellular BSs, GPS satellites, IMUs, etc.). While the latter can be applied to our target environment, they either offer less-than-desirable accuracies (e.g. tens of meters with cellular BSs), or are not functional indoors (e.g. GPS). Inertial sensors carried by responders are a possibility, but suffer from poor accuracy as well ($\approx 10-50$ m, due to drift over time), without periodic calibration and resetting to known indoor reference points. Further, the lack of access to multiple stationary APs/BSs, prevents these solutions from accurately tracking *mobile* responders in real-time. Hence, notwithstanding the plethora of prior solutions, our target environment requires a new, robust, (indoor) *infrastructure-free* solution that can *accurately* (≈1–2 m) track *mobile* responders in unknown, indoor environments from outside.


Figure 1: *Left*: The TrackIO setup with a DJI Phantom 4 UAV carrying a master UWB node and the Raspberry Pi controller units, *Right*: 4 UWB equipped helmets for first responders.

**Key design choices:** This motivates us to design a localization-tracking system from scratch, paving the way for two key design choices: (i) modality of localization, and (ii) wireless technology for localization. The lack of indoor infrastructure support, and the need to quickly deploy and localize responders in 3D, across multiple floors of a building (from outside), makes UAV (unmanned aerial vehicle) an ideal platform for the task. The UAV can serve as a virtual mobile infrastructure that is deployed on-demand, outside the building to localize the responders inside. For the choice of wireless technology, we summarize their pros-cons in Table 1. We refer to only techniques that allow long distance localization that would be applicable in our application. While lower frequencies (e.g. LTE) offer better indoor penetration/coverage (e.g. 1 Km) from outside, they are limited by available bandwidths (tens of MHz) and hence accuracy (tens of meters [35]). Higher frequencies (e.g. mmWave, $>20$ GHz) offer high accuracies (tens of cm) from large (GHz) bandwidths, but suffer from high attenuation (does not work with blockages, not accounted for in Table 1). Ultra-wideband (UWB) operates in $3-10$ GHz and offers a 1 GHz bandwidth, thereby striking a good balance between accuracy (tens of cm) and indoor penetration (tens of meters). Further, its low power design accompanied by a standardized high-resolution, ranging protocol between peer UWB nodes, makes it a synergistic choice for deployment on the UAV. Thus, our objective is *to localize and track responders (carrying UWB nodes) in*

*The work was performed during an internship at NEC Labs America.

| | LTE [35] | WiFi [25] | UWB [43] | mmW [36] |
|---|---|---|---|---|
| Accu | $>$ 20m | 5m | 10cm | 1cm |
| Range | $>$ 1km | 100m | 50m | 40m |

Table 1: UWB offers the best tradeoff between accuracy and penetrability among all RF localization technologies.

*real-time even if they are deep indoors, with the help of a UAV (also carrying a UWB node) flying outside.*

**Challenges:** One might wonder if deploying multiple UAVs to effectively serve as stationary BSs/APs outside can help solve the indoor tracking problem. We argue in Section 2.1 that having multiple UAVs outside does not guarantee access to multiple (three or more) of them by a given indoor responder, not to mention the need for their synchronization. More importantly, we show that when available, multiple UAVs need to be efficiently deployed to cover (localize) responders in different sections of the building simultaneously, rather than to serve as stationary APs/BSs. Thus, at the core of our problem, we must *localize and track indoor responders in real-time by a single UAV using its key degree of freedom, namely mobility*. This in turn poses several non-trivial challenges. *(i) Mobility of responders:* The mobility of the UAV is used to create a synthetic aperture *over time*, which serves to provide reference points for localizing an indoor node[1] through multi-lateration. However, such temporal dependency, makes multi-lateration approaches fail significantly (accuracy of about 10m, Section 3), when the indoor node is also mobile. *(ii) Indoor coverage:* While UWB's penetration capabilities are better than mmWave, they are still limited to tens of meters and hence cannot guarantee reachability to all nodes. While deploying multiple UAVs, outside different sections of the building can alleviate coverage issues, it still cannot ensure reachability to those that are deep inside the building. *(iii) Real-time tracking:* The UWB protocol provides the basic two-way ranging primitive between a UWB node-pair (UAV and responder in our case). However, employing its TDMA operational structure to collect sufficient ranging measurements to all UWB nodes from the UAV will not be scalable for real-time tracking in a large network. *(iv) Absolute location fix:* Since the UAV localizes the responders with respect to its own position, to get their absolute location fix, we need to accurately estimate the UAV's position as well. Whereas high-end UAVs employ multiple GPS receivers along with inertial sensor fusion to provide position accuracy to under a meter, lower-end UAVs provide accuracies of only around 2-3 m, thereby limiting the accuracy of the overall system.

**TrackIO:** Towards addressing these challenges, we build TrackIO – a UAV-UWB based system that is capable of localizing and tracking mobile responders to within 1-2 m accuracy from a single UAV outside in real-time, even in deep indoor environments. When multiple UAVs are available, TrackIO deploys them on different sections of the building for wider, simultaneous coverage. In realizing this, TrackIO incorporates four novel elements in its design.

*(i) Trajectory Tracking:* TrackIO adopts a first-principles approach to directly estimate the trajectory of the mobile node, rather than just its location. TrackIO analytically instruments multi-lateration formulation to not only estimate the location but also the velocity of the responder. It incorporates intelligent mechanisms for adaptively varying the size and choice of the synthetic aperture (anchor points used for localization) to address responders with non-uniform velocity (e.g. those turning corners, etc.).

*(ii) Multi-hop Localization Paradigm:* TrackIO enables a multi-hop localization paradigm for extended indoor coverage, where, responders directly reachable from the UAV ($\texttt{hop}_1$), are localized first. Then, they serve as anchors for localizing nodes ($\texttt{hop}_2$) that are reachable by them but not by the UAV. Nodes are able to dynamically estimate their own hop status based on their reachability to the UAV and overheard ranging messages from neighboring nodes. TrackIO alleviates the deterioration in accuracy over hops (due to iterative localization), by selecting only upstream nodes with accurate location estimates as anchors for downstream localization.

*(iii) Concurrent Ranging Protocol:* To enable real-time tracking even for a large, multi-hop network of nodes (e.g. big buildings), TrackIO transforms UWB's sequential ranging protocol into an *efficient, concurrent* one. It leverages the broadcast nature of the wireless medium to (a) parallelize the ranging measurements within each hop, and (b) efficiently multiplexes ranging measurements between hops,while also eliminating redundant message transmissions. TrackIO achieves a 3$\times$ speed-up, resulting in a location update frequency of 6 Hz that allows for real-time tracking.

*(iv) Reverse Location Look-up:* Instead of the UAV serving as the anchor, TrackIO now estimates the location of the UAV itself, by leveraging UWB again. It accomplishes this by using four static UWB beacons, deployed on the roof corners of a responder service vehicle, as anchors. One of these UWB beacons is also fitted with a GPS receiver, whose stationary estimates over time are highly accurate. This coupled with known inter-beacon distances, allows for accurate localization of the UAV to within a meter despite mobility.

**TrackIO's performance:** We have built a complete version of TrackIO using a DJI Phantom 4 [13] as the UAV, and Decawave DW1000 [11] as the UWB node. The ranging estimates collected at the UAV are transferred to a ground service vehicle, where TrackIO's algorithms estimate the position and trajectory of all the responders in real-time. Our real-world deployment and evaluation across multiple floors of a mid-size office building (2500 sq.m.) reveal that TrackIO is able to track indoor static nodes with a median accuracy of about 1–1.5 m and mobile (even running) nodes with a median accuracy of 2–2.5 m. A demo of TrackIO is available at http://www.nec-labs.com/trackio.

[1] Responders are synonymously referred to as nodes.

**Broader applicability:** While TrackIO leverages UAV and UWB as its modality for enabling real-time tracking of first responders, we would like to note that TrackIO's core mechanisms of trajectory tracking and multi-hop localization can be equally applicable to other localization modalities (e.g. WiFi) as well. Hence, TrackIO's contributions can also benefit other potential indoor localization and tracking applications.

## 2 Challenges in Building a Practical System

The UAV flies outside to create a synthetic aperture of anchor points, from where it ranges with each of the indoor nodes using UWB, thereby allowing for their subsequent localization through multi-lateration. Albeit straight-forward in principle, realizing this in practice faces several challenges, some fundamental, and others practical that we now outline.

**Brief Primer on UWB Ranging:** UWB nodes employ a protocol known as *two way ranging* (TWR) to estimate the distance between each other. This standard protocol [20], involves exchanging a specific set of messages (Figure 2) that cancels the effect of clock offsets between nodes. Performed in hardware with precise clocks and coupled with a 1GHz wireless bandwidth, TWR allows accurate time-of-flight estimates (even in presence of multi-path), resulting in accurate ranging (≈10 cm). One TWR exchange takes around 16.7 ms on a widely used UWB chip (DW1000 [10]).

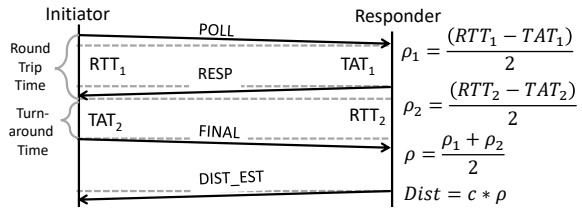


Figure 2: The original 802.15.4 TWR protocol is designed for ranging between two devices.

### 2.1 Impact of Responder Mobility

Fig. 3(a) shows a typical synthetic aperture where three representative $< location, range >$ tuples are chosen to solve for a stationary node's location using trilateration. The solution is reasonably accurate as all the three $< location, range >$ tuples are consistent with respect to a unique location of the stationary node. Contrast this with figure 3(b), where the node moves with an uniform velocity. In this case, the three $< location, range >$ tuples are no longer consistent with respect to any particular node location. Different portions of the synthetic aperture now correspond to different node locations. In other words, the node has changed its position significantly by the time the UAV started and completed building the aperture. This can affect localization accuracy by as much as 10 m, as shown in Fig. 9.

**Can we alleviate the impact of mobility?** A natural approach is to figure out if multiple $< location, range >$ tuples can be gathered from distinct UAV locations "simultaneously".

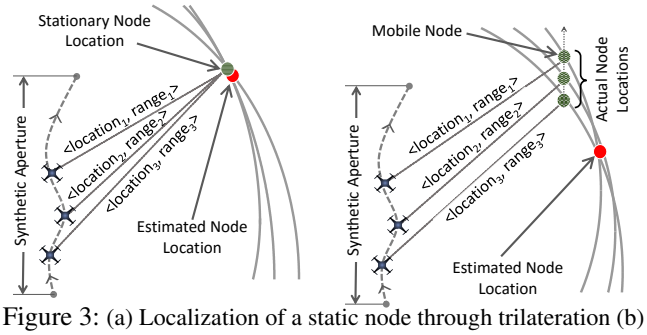


Figure 3: (a) Localization of a static node through trilateration (b) Naive trilateration fails for mobile nodes

*Using multiple UAVs:* Multiple UAVs form a spatial aperture that can simultaneously collect range estimates in principle. This is however, difficult to realize in practice for the following reasons: (a) It is unlikely that a particular indoor node is simultaneously reachable from multiple UAV locations, (b) Synchronizing the different UAVs as well as their corresponding range estimates in real-time becomes extremely challenging, and (c) Operating multiple UAVs in close vicinity requires sophisticated path planning to be done apriori. Multiple UAVs have a role to play in the broader system (for improving building coverage, as we discuss later). However, they are less useful to solve the problem of node mobility, which motivates us to address the problem with a single UAV.

*Increasing the UAV's speed relative to responders:* Another approach to counter node mobility can be to increase the UAV's velocity. Figure 4 shows the limited benefit of moving the UAV faster. Even when the UAV is traveling at 10m/s it cannot completely compensate for the node's mobility. Moreover, moving the UAV too fast causes the channel to change very rapidly, resulting in ranging errors.

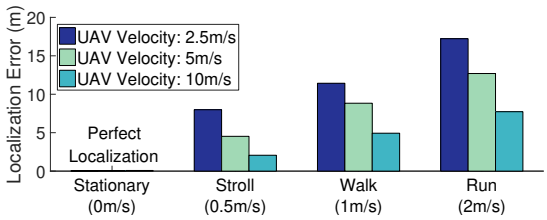


Figure 4: Localization accuracy improves with UAV speed, yet falls short of the target accuracy.

### 2.2 Insufficient Building Coverage

The FCC power emission limit for UWB transmitters is –41.3 dBm/MHz [20] that severely restricts the communication range between two UWB nodes. With the UAV located outside the building and limited indoor penetrability, some nodes that are relatively deep indoors are not directly reachable. We perform elaborate measurement studies to characterize the communication range in such indoor environments. Figure 5 shows the packet-loss percentage as the distance between the nodes increases in a cluttered indoor space. Such ranges could vary from about 30 m (50% loss) in very dense/cluttered indoor environments (e.g., rooms with concrete walls) to about

60 m in relatively open indoor spaces (e.g., office, library, shopping malls etc.).

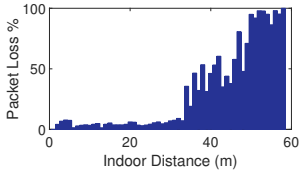


Figure 5: UWB packet-loss over various distances in a cluttered office environment.

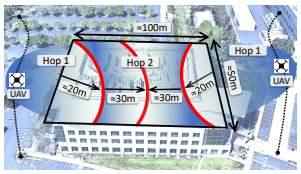


Figure 6: UAVs on the outside may not be able to cover nodes deep indoors. A multi-hop solution is necessary.

**Multiple UAVs can improve but not solve the coverage problem.** With the UAV flying approximately 10 m away from the face of the building, nodes that are about 30 m inside are directly reachable. This limits the indoor coverage area to a great extent. Note that even flying multiple UAVs along the four faces of a medium-sized building (floor area ≈10000 sq.m) only improves coverage in the building's periphery but not in the deep interiors that account for about 20% of the indoor space (see figure 6). Given the criticality of the application, complete indoor coverage is of paramount importance. This necessitates the design of a multi-hop localization network, where range estimates and hence localization can be achieved from nodes of a given hop to nodes belonging to the next hop and so on.

**Challenges with multi-hop localization.** Realizing multi-hop localization is challenging for several reasons: (i) iterative localization leads to cascading errors and hence poor accuracy across hops; (ii) nodes need to identify their reachability status (e.g $\texttt{hop}_1$, $\texttt{hop}_2$, etc.) to other nodes to help track a dynamic, multi-hop topology; (iii) orchestration of ranging measurements across hops becomes critical for ensuring real-time tracking of the multi-hop network.

## 2.3 Inability to Track Real-time

In a large network of nodes spanning multiple hops, a *time division* (TDMA) scheme needs to be designed that runs TWR across relevant pairs of nodes to estimate their range fast enough to relatively localize all nodes in the network. Clearly, executing a TWR across all pairs of nodes is not suitable: unreachable links will waste time, and in a size $N$ network, one round of (range) data collection will require $O(N^2)$ time slots. Hence, for a network consisting of several tens of nodes, collecting a single set of range information might take several hundreds of milliseconds. With several such sets needed to position the indoor nodes, the total delay can be several seconds. Further, with the mobility of nodes resulting in highly dynamic topologies, it becomes very challenging to track nodes in real-time with such an update rate of measurements.

## 2.4 Imprecise UAV Localization

Note that the UAV's location measurements need to be as precise as possible in order to leverage the highly precise range estimations offered by the UWB technology. Unfortunately, UAV location estimates obtained out-of-the-box is at least an order of magnitude less precise compared to UWB ranges. For instance, consider the location estimates obtained from a GPS device. In an open field, such locations have minimal errors (≈ 2–3m). However, in scenarios, where the UAV moves along the periphery of a building, the GPS signal reception can be significantly hampered resulting in the error to escalate to as high as 15–20m [50].

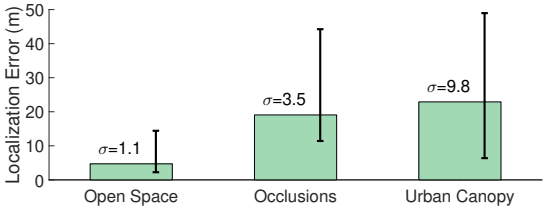


Figure 7: Effect of UAV's GPS errors on localization accuracy

In figure 7 we show the impact on localization accuracy of a static node using simple trilateration in three different deployment settings; from a relatively open space to locations having partial occlusions and urban canopies. Note that even for a static outdoor node, slightly erroneous GPS locations of the UAV can be detrimental for its eventual localization. Assuming GPS corrections and inertial sensor fusion applied by the UAV, the errors could be at best, $1-2$ m even when the node is outside and static; localizing a mobile node indoors would only lead to significantly degraded accuracies.

# 3 System Design

We now present TrackIO – a UAV-UWB based system that is capable of localizing and *tracking mobile* responders to within 1-2 m accuracy from a single UAV outside in *real-time*, even in *deep indoor* environments. TrackIO accomplishes this without the necessity or dependence of any infrastructure deployed indoors. TrackIO can almost instantly be functional from the time of launch (under a minute). This is achieved by employing a host of algorithmic and architectural changes to the underlying multilateration and ranging protocols.

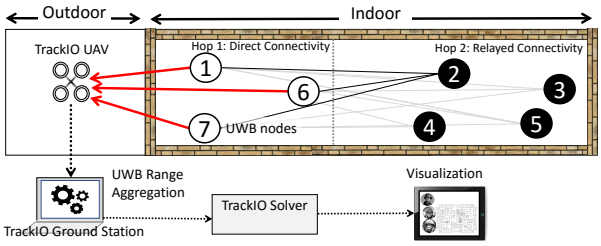


Figure 8: High level system design

### 3.0.1 Overview

Fig. 8 shows a snapshot of TrackIO in action along with its various architectural components. The UAV flying outside the building's periphery is equipped with a UWB *master node* that collects range information from the *client nodes* inside the building. The nodes are possibly worn by personnels (e.g.,

firefighters, military troops, emergency responders etc.) who are tracked through our system. Client nodes that are directly reachable from the UAV's master node are designated as $\texttt{hop}_1$ nodes, additional nodes are referred to as $\texttt{hop}_2$ nodes, $\texttt{hop}_3$ nodes and so on based on subsequent reachability. In Fig. 8, the UAV directly ranges the nodes in $\texttt{hop}_1$, who in turn range the nodes in $\texttt{hop}_2$ and relay information back to the UAV. The UAV offloads range information to a ground control station that solves for the locations of all client nodes. Additionally, we develop a mobile application that can be used to visualize the tracking information on a map with sub-second latency.

When multiple UAVs are available, they are deployed on different sections of the building and/or at different altitudes of the same section (for tall buildings) for wider, simultaneous coverage. Since each UAV would execute TrackIO in parallel, we focus on a single UAV's operation in the rest of this section. Also, for easier exposition, we focus on the UAV localizing responders in a single floor (horizontal plane) by fixing its altitude appropriately. How the UAV scans floors and identifies the appropriate altitude ($z^*$) is covered in Section 3.5. Some results presented in this section are obtained from simulation studies, which are intended for highlighting the intricate aspects of our system design. Nonetheless, sections 4 and 5 present extensive evaluation results from experiments carried out in real testbeds.

## 3.1 Tracking Trajectory of Mobile Nodes

### 3.1.1 Estimating Velocity through Synthetic Apertrure

Recall that when a node is mobile, the $< location,\ range >$ tuples measured by the UAV do not uniquely map to a single location, resulting in poor localization accuracy of multilateration solvers. Instead of alleviating the impact of mobility, TrackIO adopts a first-principles approach to directly estimate the trajectory (speed and heading) of the mobile node, rather than just its position. To accomplish this, TrackIO analytically instruments the multilateration formulation to estimate both the initial location $(x,y)$ as well as the velocity vector $(V_x\hat{i}+V_y\hat{j})$, where $\hat{i}$ and $\hat{j}$ are unit vectors along positive $X$ and $Y$ axes respectively of the node. Using these, the node's traversed path can be traced. This assumes that human mobility can be approximated with uniform velocity, which is reasonable within the short time-scales (few seconds) of the UAV's synthetic aperture. This assumption is relaxed in Section 3.1.2, where we show how non-uniform mobility (e.g. turning corners, etc.) can also be addressed in this framework.

Suppose we have ranging measurements from $n$ consecutive UAV locations – $n$ is called the aperture size and is essentially a moving window of $n$ historical measurements. For any time instant $T_i$ ($i = [1..n]$), the UAV records the mapping $< location_i,\ range_i >$, where $location_i$ is the UAV's 3D location and $range_i$ is the distance estimate of the mobile node from the UAV. The mobile node is located at an unknown location $(x_i,\ y_i,\ z_i)$. We denote the UAV's 3D-location as $(Cx_i,\ Cy_i,\ Cz_i)$. The measured range is given by:

$$range_i = \sqrt{(Cx_i - x_i)^2 + (Cy_i - y_i)^2 + (Cz_i - z_i)^2} \qquad (1)$$

Assuming we know which building floor the responder is currently occupying, we do not need to solve for $z_i$ ($= z^*$). Yet, this is a single equation with two unknowns—we cannot directly solve for $(x_i,\ y_i)$. Even if we obtain multiple such ranges, each equation will add a new set of unknowns. However, the new unknowns are not independent, but related through the node's velocity. Hence, assuming the node is moving at a constant velocity, there are inherently only four unknowns $(x_1^*,\ y_1^*,\ V_x^*,\ V_y^*)$ that do not increase with additional ranges, thereby allowing us to solve for them.

We can reformulate this as an unconstrained minimization problem that attempts to find the best fit, i.e. location and velocity that minimize the following error function:

$$(x_1^*,\ y_1^*,\ V_x^*,\ V_y^*) = \arg \min_{(x_1,\ y_1,\ V_x,\ V_y)} f$$

$$f = \sum_{i=1}^{n} ((Cx_i - x_i)^2 + (Cy_i - y_i)^2 + (Cz_i - z_i)^2 - range_i^2)^2 \qquad (2)$$

The various $(x_i,\ y_i)$ are obtained from the intial $(x_1,\ y_1)$ and velocity $(V_x, V_y)$ based on kinematic equations:

$$x_i = x_{i-1} + V_x \cdot \Delta T_i = x_1 + V_x \sum_{j=1}^{i} \Delta T_j$$

$$y_i = y_{i-1} + V_y \cdot \Delta T_i = y_1 + V_y \sum_{j=1}^{i} \Delta T_j \qquad (3)$$

where $\Delta T_i$ denotes the time between measurements. Since $(x_i,\ y_i)$ are generated based on the initial location $(x_1,\ y_1)$ *and* the velocity vectors $(V_X,\ V_y)$, by minimizing equation 2, we obtain the closest approximation of both, location and velocity vectors for the node. The first output from this solver is obtained only after $n$ measurements (typically a few seconds worth of data) have been recorded. Thereafter, a location update is obtained for every round of range measurements. Thus, the system's steady state update rate depends only on the duration of one range measurement round, and does not depend on the aperture size.

We now analyze the improvement in localization achieved by incorporating velocity vectors over simple multilateration. Our simulation framework mimics a UAV and a set of indoor UWB nodes that follow predetermined trajectories at any desired speed. We introduce an empirically derived range estimation error to the ranges. Fig. 9 shows how simple multilateration results in higher localization errors with increasing node velocities. The UAV is assumed to move at a fixed 5 m/sec velocity. Note that even for human walking speeds the error could be as high as 10 m. On the contrary, the velocity-based solver is least impacted by increasing velocity of the mobile node.

### 3.1.2 Adaptive Apertures for Non-uniform Velocity

The above approach assumes that the node does not change its velocity (speed or direction) during the course of one aperture window (say, 4 secs). However, this assumption is broken if

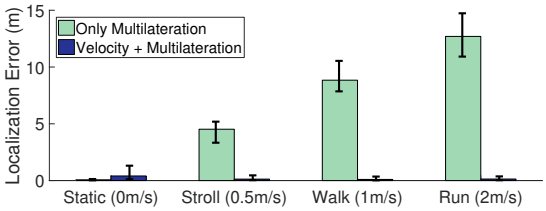


Figure 9: Localization error remains minimal when using velocity vectors even under fast human mobility

the node turns, accelerates or halts. In principle, this could be solved by adding higher order derivatives of the node's location (e.g., acceleration, jerk) into the kinematics equations employed by our location solver. However, our analysis indicates that such an approach is rather contrived. It makes the solver prone to overfitting and extremely sensitive to range errors. Further, given the short time scale of the aperture window, we find that the approximation of uniform velocity does not hurt the performance much during acceleration and halting but does induce significant errors during *turns*, which we now address. We propose to utilize the solver's confidence in the estimated location to infer non-uniform velocity and when detected, trigger an aperture reset that eliminates measurements prior to the turn.

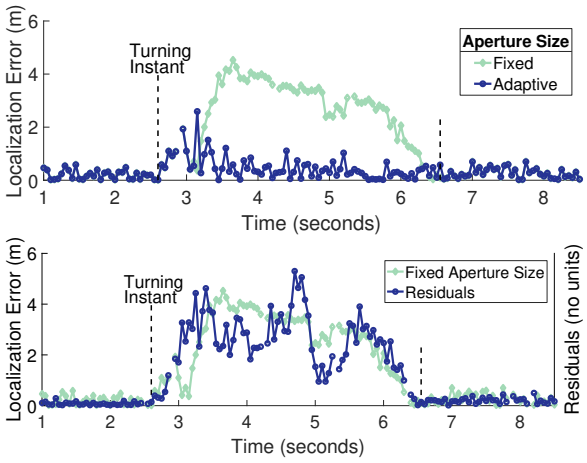


Figure 10: (a) Localization error is high during turns. Resetting the aperture helps curtail the loss in localization accuracy. (b) The high error-residual also indicates low solver confidence in the location estimate providing a hint for turn detection.

**Impact of turns.** Fig. 10(a) shows the impact of sudden turns on the localization error (green line). We simulated a node moving in a straight line, then taking a 90° turn, and continuing again in a straight line. An aperture of 4 seconds—UAV locations and the corresponding ranges of the past 4 seconds—are used to estimate the current node location. Observe how the localization error (grey line) starts to increase from the point where the aperture's *head* crosses the turning position (first dashed vertical line) and falls back to its pre-turn values after the aperture's *tail* has crossed the turning position (second dashed vertical line).

**Adaptive aperture to address turns.** If we have a mechanism to detect turns, we could potentially eliminate historic measurements till the turn and restart constructing the aperture. To understand the benefit, we introduce the notion of an adaptive aperture in the above simulation. At the time of the turn, we remove all history and restart estimating location after a short history has built up[2]. Just after resetting history, the localization error is indeed high (dark blue line just after the "turning instant" in Fig. 10(a)) but quickly recovers and becomes acceptable once the aperture fills up with relevant measurements after the turn. In comparison, if a fixed aperture size is used, the effects of a turn last for the entire duration of the aperture (green line).

**Triggering an adaptive aperture.** During turns, the solver is unable to provide a reasonable answer since no single velocity estimate can represent all the measurements. This results in larger residual errors after solving Equation 2. Observe in Fig. 10(b) that the solver's residuals (in arbitrary units) are highly correlated with localization error. Thus, a sudden increase in the residuals helps identify non-uniform velocity events such as turns. We use Gaussian Mixture Models on the residuals to identify a changing trend in them and captures such events.

In summary, localization of mobile nodes, even those with non-uniform velocity is possible through a combination of joint location-velocity solving and by adaptively resetting the aperture size. At any given instant, our solver uses different aperture sizes that are appropriate for each node.

## 3.2 Multi-hop Tracking for Coverage

TrackIO is designed to function even if some nodes are beyond the UAV's direct range. TrackIO allows such unreachable nodes to range with other nodes in the vicinity which can in-turn reach the UAV and/or have already been localized. Thus, a multi-hop topology is dynamically created with nodes belonging to different hops based on their reachability characteristics. The UAV's synthetic aperture localizes first hop (directly reachable) nodes. These $\texttt{hop}_1$ nodes then act as anchors for localizing $\texttt{hop}_2$ nodes. This process iterates across hops. TrackIO employs several mechanisms to ensure that mobile nodes can be accurately localized even across multiple hops.

**Dynamic estimation of hop membership.** Nodes that are within the UAV's UWB communication range, directly receive ranging messages initiated by the UAV, and classify themselves as $\texttt{hop}_1$ nodes. Those that do not receive messages from the UAV but receive some of the response messages sent by $\texttt{hop}_1$ nodes, classify themselves as $\texttt{hop}_2$ nodes and so on. Thus, nodes can determine their own hop membership in a decentralized manner.

**Anchor selection for iterative localization.** Two components contribute to the final localization error of $\texttt{hop}_\texttt{m}$ nodes:

[2]During the short period that new history is being built, the system continues to output results from the previous aperture.

1. relative localization error of $\text{hop}_m$ nodes with respect to $\text{hop}_{m-1}$ nodes, and 2. localization error of $\text{hop}_{m-1}$ nodes chosen as anchors. Without loss of generality, $\text{hop}_{m-1}$ nodes can be assumed to be spaced far apart compared to the synthetic aperture formed by the UAV. This increased spacing between anchors, *improves* $\text{hop}_m$ localization, compared to that of $\text{hop}_1$ nodes (w.r.t. the UAV). However, the localization error of the $\text{hop}_{m-1}$ nodes and recursively that of upstream hop nodes, will cumulatively contribute to the error of $\text{hop}_m$ nodes. Thus, the choice of anchors in $\text{hop}_{m-1}$, has a cascading impact on the localization accuracy of downstream nodes (i.e. hops $\geq m$).

Nodes that are static or moving with a uniform velocity in $\text{hop}_1$ inherently have better localization accuracy than those with non-uniform velocity. Hence, by leveraging the solver's ability to identify such nodes (those with high residuals), TrackIO avoids selecting them as anchors for localizing $\text{hop}_m$ nodes, curtailing the cascading effect across hops.

**Instantaneous mobility tracking beyond $\text{hop}_1$.** In contrast to the first hop nodes which are localized through a *temporal* aperture created by the UAV's motion, $\text{hop}_m$ ($m > 1$) nodes are localized through a *spatial* aperture formed from a diverse placement of $\text{hop}_{m-1}$ nodes. This decoupling (from UAV's mobility), allows for *instantaneous* localization of $\text{hop}_m$ nodes from previously obtained $\text{hop}_{m-1}$ locations. The time scale of such localization is in milliseconds within which the nodes move a negligible distance. As a result of the spatial aperture employed, $\text{hop}_m$ ($m > 1$) nodes can use conventional multilateration approaches (without need for velocity vectors) even when they are mobile.

**Localizing $\text{hop}_1$ nodes with non-uniform velocity using downstream spatial apertures.** Unlike $\text{hop}_1$ nodes, mobility is not a concern for $\text{hop}_m$ ($m > 1$) nodes as they are instantaneously localized using a *spatial* aperture formed from high-confidence (low residual) $\text{hop}_{m-1}$ nodes. Hence, $\text{hop}_2$ nodes can in-turn, form a spatial aperture (serve as anchors) and correct the location of $\text{hop}_1$ nodes, which are currently experiencing non-uniform velocity (low confidence, high residual). Fig. 11 shows the localization accuracy of a turning $\text{hop}_1$ node using ranges from $\text{hop}_2$ nodes. Observe how the turn gets localized precisely using the spatial aperture from $\text{hop}_2$. Thus, TrackIO is able to eliminate most of the impact of non-uniform velocity of $\text{hop}_1$ nodes.

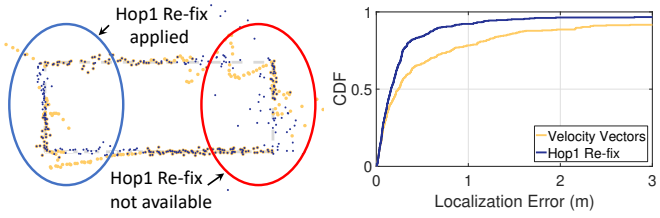


Figure 11: Re-fixing $\text{hop}_1$ nodes using $\text{hop}_2$ nodes improves localization even during turns. But, it may not be always available.

Note that this downstream spatial aperture technique is opportunistic – it can be used when enough $\text{hop}_2$ nodes exist. In contrast, the mechanism of adaptive (temporal) aperture of the UAV described in Section 3.1, provides benefits even when no other nodes exist in the topology. Therefore, TrackIO incorporates both these techniques to address non-uniform velocity of $\text{hop}_1$ nodes.

**Leveraging multiple UAVs vs. multiple hops.** While multi-hop localization allows for coverage of even large buildings using a single UAV, the localization error of its downstream hop nodes (i.e. $m > 2$) will increase and might not satisfy our desired target of 1-2m. Hence, TrackIO leverages the multi-hop paradigm to primarily reach deep interiors of buildings (where even multiple UAVs cannot help), while employing multiple UAVs to provide non-overlapping, peripheral coverage for large buildings. Thus, using a combination of multiple UAVs and hops, TrackIO is able to cover large buildings with just two hops from a single UAV.

**Handling $\text{hop}_2$ disconnections.** In rare circumstances, if a node goes out of range from all hop-1 nodes, its localization must rely on an IMU-based dead reckoning system. This approximate location estimate will then be communicated with the UAV using alternative communication modes (such as WiFi or cellular data). Adding support for such eventualities is left to future work.

## 3.3 Concurrent Ranging

A fast and reliable ranging protocol is essential to create a real-time localization system. Since the UWB ranging protocol is designed for ranging between a pair of nodes, it does not broadcast messages. This leads to a sequential ranging of every node in a hop, which is not scalable for real-time operation, especially in a multi-hop network. The key idea in TrackIO is to leverage the broadcast nature of wireless signals to communicate and hence *concurrently* range with multiple nodes using a single transmission. To this end, TrackIO makes appropriate modifications to the underlying protocol (Fig. 2) to create a concurrent ranging scheme (Fig. 12). We describe the scheme for the first two hops; subsequent hops are similar.

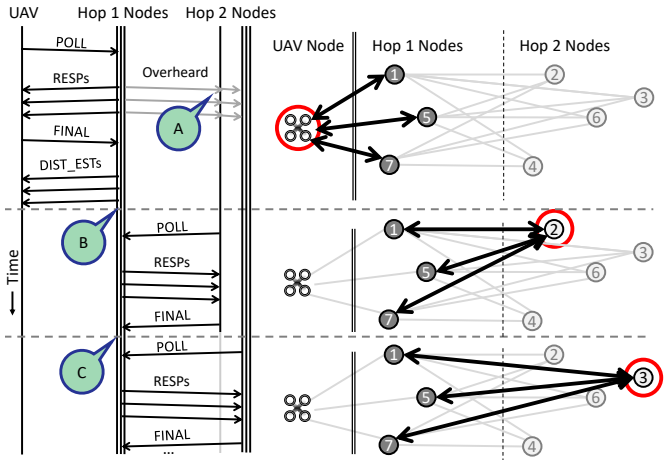


Figure 12: Progress of the protocol in a 2-hop example topology.

**Concurrent ranging at $\text{hop}_1$.** The UAV *simultaneously* ini-

tiates ranging with all reachable $hop_1$ nodes by broadcasting a single POLL. Each node that receives this message, takes turns (based on its hard-coded NodeID) to send a RESP message. After collecting the timings from all the RESPs, the UAV broadcasts a single FINAL message containing information for all $hop_1$ nodes. On receiving the FINAL, all $hop_1$ nodes calculate their distance from the UAV and send it back to the UAV (DIST_EST messages).

**Concurrent ranging at $hop_2$.** Identical to $hop_1$ nodes, $hop_2$ nodes listen to the channel for messages. However, being outside the direct communication range of the UAV, they cannot receive the POLL message. Instead, they only overhear the messages sent by nearby $hop_1$ nodes in response to the UAV's POLL (point A in Fig. 12). After all $hop_1$ nodes have completed sending their DIST_EST messages (point B in Fig. 12), the first $hop_2$ node initiates a full sequence of POLL-RESPs-FINAL *simultaneously* with all $hop_1$ nodes in the vicinity. $hop_2$ nodes follow the same protocol as the UAV with one subtle difference. $hop_1$ nodes do not send DIST_ESTs back to $hop_2$ nodes. Instead, $hop_1$ nodes calculate and locally store all the $hop_1$-$hop_2$ ranges, which are piggybacked on the subsequent DIST_EST message. This saves unnecessary network overhead, speeding up the collection of range estimates. All $hop_2$ nodes take turns (point C in Fig. 12), followed by the UAV starting the next round.

**Efficient multiplexing of ranging between hops.** Initially the UAV is not aware of the topology. Hence, it waits for all the nodes in the network to send a RESP. Once it has received the last RESP (or, after a timeout), the UAV creates a bitmap (Fig. 13) indicating which nodes are deemed to be in $hop_1$ (setting the corresponding bit to one) based on the responses. The UAV sends this bitmap in its FINAL message. When $hop_1$ nodes send a DIST_EST message it also contains a copy of this bitmap. A node that receives such a DIST_EST, but not the POLL from the UAV, would see its bit cleared and know that it belongs to hop2. Also, it would know how many other $hop_1$ nodes are expected to send their DIST_ESTs, and the order of all other $hop_2$ nodes. This allows $hop_2$ nodes to efficiently take transmission turns without collision, even when they are not in communication range of each other and the UAV. The UAV generates the bitmap dynamically in every round to track topology dynamics due to node mobility.

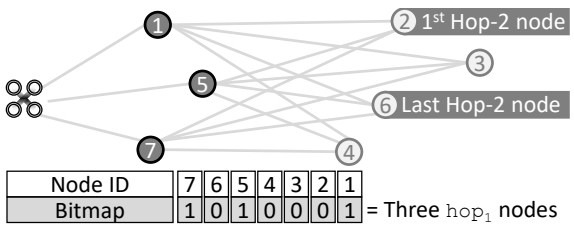


Figure 13: The bitmap constructed by the UAV and sent in the FINAL message enables a collision free $hop_2$ communication.

Finally, the variable length DIST_EST sent by $hop_1$ nodes piggybacks their distance from all $hop_2$ nodes obtained in the *previous* round, along with their own *current* range estimates to the UAV. The UAV aggregates all the information received in the DIST_ESTs and forwards to a ground control center for further processing.

## 3.4 Reverse Lookup for UAV Location Fix

Obtaining the UAV's precise GPS location is critical to TrackIO's end-end accuracy. This can be challenging since off-the-shelf GPS receivers have multi-meter location errors [24, 50]. High-End UAVs already employ GPS chips with better precision and higher update rate [14], and improve the precision further by incorporating IMU data as well. Some UAVs [14] also support custom, albeit expensive RTK solutions [39] that promise location accuracy within a few cm, but require precise GPS transmitters in the vicinity.

In cases, where such precise UAV location estimates are not possible, TrackIO leverages UWB to also localize the UAV. It places four *static* UWB nodes as anchors at known locations on the ground. One of these anchors is also fitted with a GPS receiver. The known, exact, pairwise distances between the anchors enables TrackIO to accurately determine the GPS coordinates of all the static anchors. These static anchors in turn allow for accurate localization of the UAV itself. We envision that these anchors can be permanently mounted at the four corners of a service vehicle (at different heights to provide vertical diversity). The service vehicle can use sophisticated GPS techniques [19, 21, 22, 26, 38, 39] to achieve better accuracy for the ground anchors.

## 3.5 TrackIO's Operations in a Nutshell

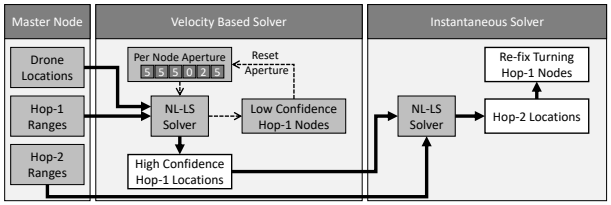


Figure 14: TrackIO System Design

When a UAV is launched to cover a section of the building, the static UWB anchors on the ground start localizing the UAV to get its precise location estimate. The UAV performs its flight trajectory to start creating a continuous, moving window of synthetic apertures (of 4 secs each). Within each of its aperture window, it performs the following. It executes its concurrent ranging protocol to help classify nodes into various hops based on their reachability. It then localizes the $hop_1$ nodes first using its location-velocity solver that estimates both the location and velocity. Using the error residuals of the solver, it employs only high-confidence $hop_1$ nodes (static or uniform velocity) as anchors for localizing the $hop_2$ nodes. For the latter, it employs a conventional multi-lateration solver to obtain only the location of $hop_2$ nodes, which being instantaneous, is sufficient. Finally, it uses a spatial aperture of $hop_2$ nodes (anchors), along with an adaptive (temporal)

aperture from the UAV, to refine the location estimate of the $\texttt{hop}_1$ nodes that have non-uniform velocity.

**Altitude Considerations.** So far, we have only focused on the horizontal plane and assumed that TrackIO is aware of the nodes' altitude. However, in cases, when TrackIO is not aware of the floor where the service personnel currently are, the horizontal localization error can be significant (since the algorithm will not take into account the additional vertical offset the signals have to travel). We address such situations by detecting the appropriate altitude (and hence floor) through a special one-time maneuver of the UAV. We move the UAV up and down through a short vertical distance that spans the target floors. During this movement, as the UAV approaches the horizontal plane of the nodes, its range estimates to the nodes should start to decrease, reach a minimum when it is on the plane, and increases when it moves away from the plane. TrackIO records the altitude ($z^*$) as that corresponding to the minimum range estimates and hence determines the floor of interest. TrackIO then uses this altitude to execute its localization process for the target floor.

## 4 Implementation and Testbed Setup

We build a custom payload consisting of a Decawave `DW1000` UWB module and a Raspberry Pi 3 used as the TrackIO controller. The payload, weighting about 200 grams is mounted onboard a DJI `Phantom 4` UAV platform. A fully charged UAV flight with our current prototype lasts for about 20 mins (≈25 mins without payload). In the following we describe the key hardware/software components that form TrackIO.

### 4.1 TrackIO Components

**UWB Modules:** The UWB module mounted on the UAV acts as the master node and is responsible for collecting ranging information from the client nodes. Alongside the DW1000 RF chip, the UWB module houses an ARM based microcontroller that runs our multi-hop ranging protocol (implemented in about 3000 lines of `C` code). The latter collects inter-node ranging information (at about 6 Hz) which is read by the controller Raspberry Pi and forwarded to the ground station through a WiFi interface.

**Ground Station:** Ground station refers to the compute node responsible for collecting ranging information obtained from the UAV and running TrackIO's localization algorithms. First, it localizes the UAV using the four fixed client nodes on the ground. These nodes are placed at different heights (vertical diversity) on four vertices of a 5 m×5 m square to emulate a service vehicle housing the ground station. One node is equipped with a GPS receiver for an absolute location fix. Second, the UAV's location is fed along with the rest of the range information that simultaneously solves all client node locations. We implement the solver algorithms in *Python* that run in real time on the ground station compute node (a Core i7 Lenovo laptop). We also implement an Android application that shows client node locations on a map.

**Flight Automation:** Automating the flight offers flexibility to programmatically control the flight's trajectory as well as its speed. We use the `Android Mobile SDK` [12] provided by DJI to program two candidate trajectories for our UAV to follow: (a) `STRAIGHT`, a straight line path of length 30 m, and (b) `WAVY`, a sinusoidal path of length 30 m with an amplitude of 5 m (see figure 17). Note that such automation also helps us to re-run/repeat flights for controlled experiments, which would have been otherwise impossible in case of manually controlled flights.

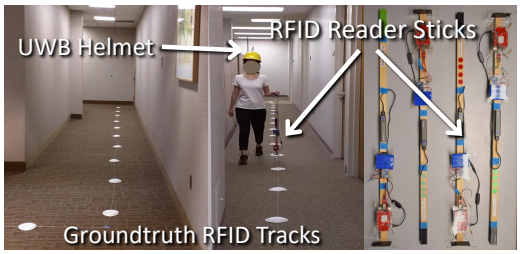


Figure 15: Snapshots of trajectories marked with RFID tags. A volunteer is shown walking along the track with the RFID reader stick in her hand.

### 4.2 Testbed Setup

We deploy TrackIO with a single UAV in the third and fourth floor of our department building spanning approximately 1250 sq. meters (half the building's floor area). Nine client nodes are placed indoors that mimic static or mobile first responders. Out of these nine nodes, six are in `hop-1` (3 static, 3 mobile) and remaining three in the `hop-2` (2 static, 1 mobile).

**Obtaining location groundtruth:** For static indoor nodes, the node location is accurately estimated with the help of a laser ranger. For tracking the mobile nodes' groundtruth positions, we deploy fixed RFID tags on the ground, one every meter, along predefined trajectories. We create portable *RFID reader sticks* equipped with a ThingMagic M6E-Nano readers (Fig. 15). We adjust the reader's transmit power as well as the antenna orientation to limit the reading range to about 50 cm. The volunteers are instructed to move along the trajectories while holding the stick vertically. The stick also hosts a Raspberry Pi that controls the reader and logs timestamped entries of RFID tags along with RSS/phase information (≈10 Hz) it reads along the trajectory. We post-process such logs to obtain accurate estimations of position (within 20 cm) and velocity of the mobile node at granular timescales.

**Trajectories:** We lay out the trajectories within our office area spanning multiple rooms, cubicles, hallways and open spaces. Specifically we construct four different trajectories, three in the first hop and one in the second hop. We create the trajectories with increasing number of turns in them. The first three trajectories are (a)`LINE`, a linear trajectory of length 20 m, (b)`TRIANGLE`, a triangular trajectory with a perimeter of 30 m and (c)`RECT`, a rectangular trajectory with a perimeter

of 40 m. The trajectory in the second hop is roughly a 30 m long sinusoidal path (SINU).

## 5 Evaluation

We present evaluation results from experiments conducted in our real testbed discussed in §4.2. Recall that we use 6 nodes (3 static, 3 mobile) in $hop_1$ and 3 nodes (2 static, 1 mobile) in $hop_2$. Four volunteers (mobile nodes) are simultaneously instructed to move along their designated trajectories at different speeds. Combined, we accumulate over 2 hours worth of traces accounting for over 10+ Kms of total trajectory length. Evaluating TrackIO's performance (w.r.t. groundtruth) through controlled experiments requires us to do trace-driven analysis of the ranging information logged by the ground station compute node. However, our system receiving range information at 6 Hz is capable of *real time* operations. In §5.4, we highlight end-to-end latency of our system for various node distributions. Fig. 16 shows the median localization error for both $hop_1$ and $hop_2$ nodes, the latter localized using static or mobile $hop_1$ nodes. While static nodes are localized with an accuracy of $1--1.5$ m, note that even for mobile nodes, the median localization error is a little less than 2 m ($hop_1$) to around 2.5 m ($hop_2$). In extreme cases, where the $hop_2$ nodes are localized using all mobile $hop_1$ nodes and the latter do not offer a good spatial diversity (e.g., all $hop_1$ nodes are in close vicinity), TrackIO still offers a localization accuracy of about 4 m (top 10 percentile). However such situations can be avoided by judiciously selecting nodes in $hop_1$ that offer spatial diversity.

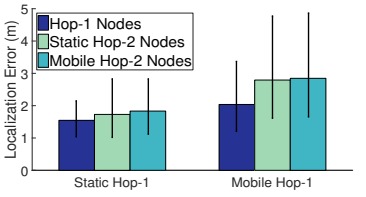


Figure 16: Overall localization accuracy for static and mobile $hop_1$ and $hop_1$ nodes

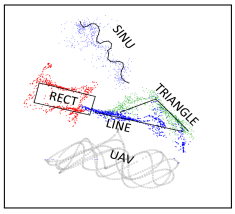


Figure 17: Scatter plot of estimated trajectories for all mobile nodes.

### 5.1 $hop_1$ Localization Performance

We now evaluate the $hop_1$ localization error over the dimensions of node speed, trajectory, and turns.

**Effect of node speed.** Since TrackIO jointly solves for both location and velocity of each node, ideally, the localization error should be independent of the velocity for nodes moving at a constant velocity. However, when moving at a brisk pace, the human body performs a complex set of movements, including bobbing of the head, which strains the constant velocity assumption. To evaluate these practical limitations, a volunteer moved along LINE at different speeds—a stroll, walk, and a run. Fig. 18 shows the resulting localization accuracy. The reported speeds are the average speed obtained from ground-truth. The median localization accuracy is around 1.5 m during the stroll, whereas it is around 2.8 m during the run. The gains over simple multilateration are above 3× for all the velocities.

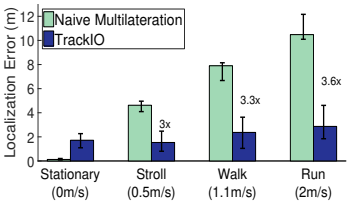


Figure 18: TrackIO gains significantly over simple multilateration by solving for velocity vectors as well.

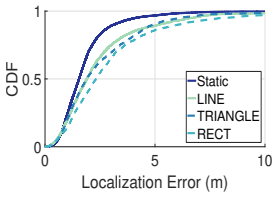


Figure 19: The three trajectories have similar localization error.

**Effect of trajectory shapes.** Fig. 19 shows the impact of different trajectory shapes (LINE, TRIANGLE, RECT) on the localization accuracy. While using naive multilateration, the localization errors can spike upto 10 m, TrackIO makes the system resilient to turns (median ≈2 m for all trajectories). Minor differences do exist which can be explained by the increasing number of turns present in the respective trajectories.

**Effect of Turns.** Human motion mostly comprises of straight lines interspersed with turns of various degrees. Fig. 20 shows the localization errors during turning events versus that during traversal in straight line segments. Note that the errors can significantly spike during such turning events (3× at 80 %ile).

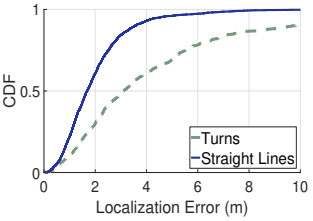


Figure 20: Localization error worsens during turns

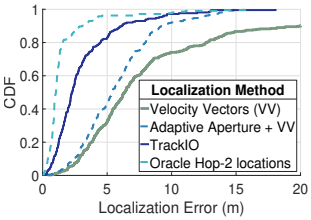


Figure 21: Benefit of spatial aperture vs. adaptive aperture

TrackIO tackle such cases through adaptively changing the aperture and spatial aperture offered by $hop_2$ node positions to fix the erring $hop_1$ nodes as described in §3. Fig. 21 shows the effectiveness of these approaches in our testbed and will be referred to in the following analysis.

**Benefit of Adaptive Aperture.** Fig. 22 shows a time series of localization error when a turning event occurs. The localization error increases sharply and remains large while the aperture slowly moves over this point in time. Our adaptive aperture dynamically resets historical measurements in case it detects turning events. It improves localization accuracy by a factor of 1.8×, and also reduces the time (*early recovery* ≈ 5 secs) it takes to stabilize the localization performance.

**Benefit of spatial aperture from $hop_2$ nodes.** Opportunistic re-fixing of a turning $hop_1$ node, $T$ might be possible if there are at least three $hop_2$ nodes that do not depend on $T$ for their localization. Fig. 21 shows the reduction in localization error when a fast moving and turning node is subjected to $hop_2$ guided re-fix. As an example, we use the entire 2 m/s

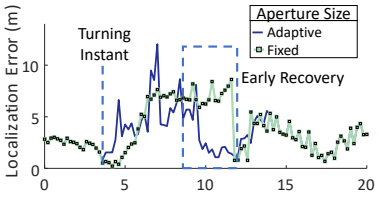


Figure 22: Adaptive aperture recovers from the effects of a turn earlier than fixed aperture.

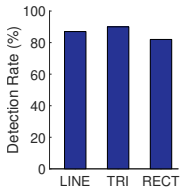


Figure 23: TrackIO correctly detects more than 80% of turns.

run shown in Fig. 18 *including the turns* at either ends. The velocity-vector based localization performs poorly as seen from Fig. 21. We use 3 other static $\texttt{hop}_1$ nodes to localize 3 static $\texttt{hop}_2$ nodes and then use their locations to solve for the location of the running node, $T$. We observe about 2.8× improvement in the median localization accuracy after re-fixing. This re-fixing accuracy is affected by two factors: (a) ranging error, and, (b) imprecise location of the $\texttt{hop}_2$ nodes. We can eliminate the effect of imprecise locations and hypothetically study just the ranging error effect by assuming ground truth $\texttt{hop}_2$ node locations are known—as if given by an Oracle. Fig. 21 shows this error to be within $2m$ at the 75%ile. While extremely promising, the re-fix approach may not always be available depending on the current topology. In comparison, the adaptive aperture technique is always available for any $\texttt{hop}_1$ node. Fig. 21 puts both these approaches (adaptive aperture and re-fix) into perspective.

**Effect of drone trajectory and velocity.** Fig. 24 shows the localization precision of two different trajectories, `STRAIGHT` and `WAVY`, at two different drone speeds. In general, geometric diversity of measurements helps obtain better localization. Therefore, the faster the drone moves, the better is the localization. Similarly, a `WAVY` pattern of drone movement also helps in obtaining better localization even for lower speeds. We therefore fly the drone in a `WAVY` pattern for this evaluation.

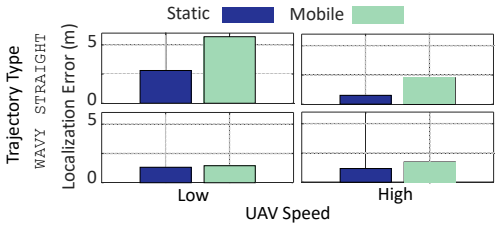


Figure 24: A `WAVY` trajectory provides better localization accuracy due to increase in spatial diversity.

## 5.2 $\texttt{Hop}_2$ Localization Performance

**Effect of $\texttt{hop}_1$ mobility.** We perform instantaneous localization[3] of $\texttt{hop}_2$ nodes based on 3 $\texttt{hop}_1$ nodes. By selecting which 3 nodes to use for this purpose, we obtain a combination of static and mobile $\texttt{hop}_1$ nodes—ranging from all static to all mobile. Fig. 25 shows the impact on localization errors as we allow an increasing number of $\texttt{hop}_1$ nodes to be mobile. These results show the error-span between using only static $\texttt{hop}_1$ nodes (1.83 m) to using only mobile $\texttt{hop}_1$ nodes (2.84 m). If multiple $\texttt{hop}_1$ nodes are available, choice of anchors influences $\texttt{hop}_2$ localization error (Fig. 26). Due to instantaneous localization of $\texttt{hop}_2$ nodes, the accuracies for both static and mobile $\texttt{hop}_2$ nodes are similar (see Fig. 16).

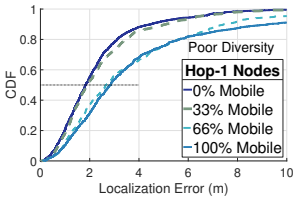


Figure 25: Localization error for a mobile $\texttt{hop}_2$ node with increasing number of mobile $\texttt{hop}_1$ nodes.

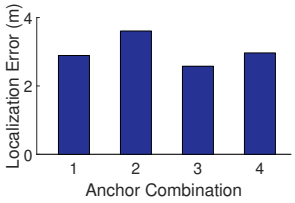


Figure 26: Selection of anchors influences localization error.

**Effect of $\texttt{hop}_1$ diversity.** The relative locations of $\texttt{hop}_1$ nodes also affect the localization accuracy. We consider random static snapshots of $\texttt{hop}_1$ node locations moving on the three trajectories (`LINE`, `TRIANGLE`, and `RECT`) and further localize $\texttt{hop}_2$ nodes. Figure 27 shows a long tail indicating that some $\texttt{hop}_1$ position combinations perform poorly. A further analysis of such failing combinations reveals that $\texttt{hop}_1$ nodes are nearly collinear[4] in such cases, causing very high dilution of precision [8, 34, 56] (Fig. 28). We expect such situations to be minimal and short lived in real-life.

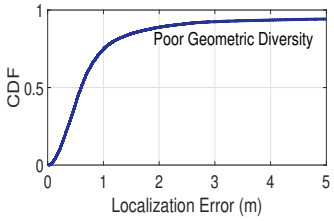


Figure 27: $\texttt{hop}_2$ localization error as a function of various $\texttt{hop}_1$ locations. Localization fails about 10% times.

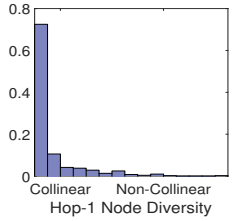


Figure 28: Most failure cases are caused by highly collinear $\texttt{hop}_1$ nodes.

## 5.3 UAV Localization

Note that the final localization accuracy of the mobile nodes is tied to absolute location fixes for the UAV. We study the impact of such accuracy as a function of the different modalities we can localize the UAV through (viz, COTS GPS receiver [1], UAV's GPS with sensor fusion [13], UWB based). Fig. 29 shows the error in each of these modalities compared to laser ranger based groundtruth (accurate to 1mm). Fig. 30 shows the improvement in localization error of a static indoor node (1 m using UWB versus 2+ m using GPS) when UWB based drone localization is used. Note that the UAV's large trajectory and the low vertical diversity in ground-UWB anchors degrades the drone's location accuracy. Yet, in GPS chal-

[3] Instantaneous localization does *not* depend on UAV's synthetic aperture created in time.

[4] We define collinearity as the ratio of the height and the base of the triangle formed by the three $\texttt{hop}_1$ nodes.

lenged situations, such as in a dense urban space, UWB based drone localization will remain valuable.

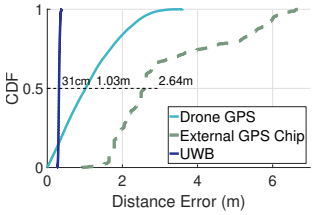


Figure 29: Range error between two fixed locations on the ground for different modalities.

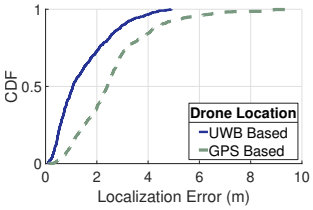


Figure 30: Effect of UAV localization modalities on static $\texttt{hop}_1$ localization accuracy.

## 5.4 TrackIO End-to-End Latency

We present TrackIO's end-to-end latency for different topologies in Figure 31. Each topology shown assumes an additional 4 $\texttt{hop}_1$ nodes on the ground for UAV localization. Thus, topology A consists of $4+6=10$ $\texttt{hop}_1$, and 3 $\texttt{hop}_2$ nodes. With even 20 nodes TrackIO leaves room for real-time operations.

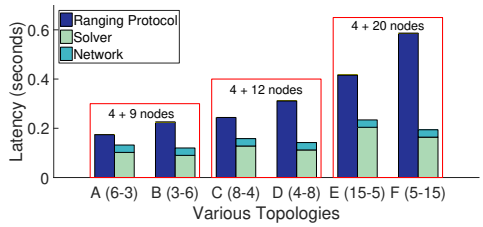


Figure 31: Protocol latency as a function of the network topology

## 5.5 Adding IMU: A What-If Analysis

Adding an IMU to our implementation might improve TrackIO's performance due to availability of another estimation for velocity and direction. We show in Fig. 32 and Fig 33 that while improvement in performance are possible when accurate direction and velocity information is available, presence of small errors in those estimates substantially reduce the gain over TrackIO. We leave more sophisticated IMU-based implementation to future work.

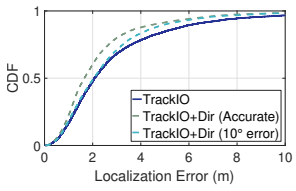


Figure 32: Improvement using direction from IMU

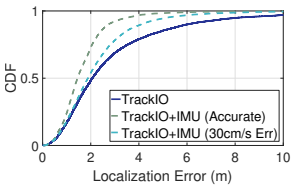


Figure 33: Improvement using velocity from IMU

# 6 Related Work

**Indoor localization.** A significant amount of work exists in indoor localization [5,33,42,51,54,55,57], but most of it relies on indoor infrastructure and fingerprinting. Both of these are not available in our target application. Techniques that use commodity WiFi [28,31] rely on the difference in subcarrier phases. However, subcarrier phase wraps after a short distance ($7.5-15$ m) rendering them unsuitable in our application. Localizing from outside a building could be performed with RF sensing [2,3], however, we require a system that would be robust to changing multipath, fast human mobility, and would penetrate deep into a building. Use of inertial sensors (IMU) for tracking human motion [42,44,52] has been extensively studied. However, most of these systems suffer from drifts and saturation introduced by the IMU sensors [18,30,46]. In contrast to IMU-based tracking, the ranging approach we take in TrackIO is not based on dead-reckoning and instead provides instantaneous location.

**UWB based localization.** UWB radios are increasingly being used for localization solutions in a variety of applications from positioning [37,40], to tracking industrial objects [17,47], to sports analytics [18]. UWB is particularly resistant to indoor multipath [41,43,45] due to its $1ns$ time resolution ($1GHz$ bandwidth). This makes it suitable for use in indoor spaces where multipath can be rampant. Different UWB platforms are commercially available today [7,10,48], and we chose Decawave Trek1000 UWB platform for its superior performance [27,43]. Most of these works assume some static UWB anchors. In our application however, we have no pre-deployed anchors, but create a synthetic aperture over time by flying an anchor on a UAV. Some recent works [32,49] have explored use of the multipath profile as virtual anchors localize using a single UWB device. However, they assume the location of all strong reflectors are known, making it prone to issues when multipath could change, due to moving people or objects. In contrast, our technique does not depend on the knowledge of the floor-plan, and is robust to changing multipath profile.

**Localization of UAV.** UAV localization has been extensively studied and approaches range from using a single GPS [13,14], to using differential GPS [19], to using complex motion models based on the drone's IMU data. UAV localization using UWB has been proposed in [6,29]. We incorporate UAV localization into TrackIO protocol. Authors of [15,23] also consider UAVs as a vehicle for fire-fighting, though they do not discuss the outdoor-indoor localization problem.

# 7 Conclusion

Indoor localization without any support from the building's infrastructure is a challenging yet important problem. Particularly of importance to first responders, continuous real-time tracking can be a life saver in many everyday situations. TrackIO uses a UAV to create the missing infrastructure *outside* the building and performs continuous ranging with indoor nodes. Through numerous algorithmic, architectural, and engineering modifications to trilateration and ranging protocols we obtain promising results localizing mobile indoor nodes accurate to about $2m$ from twenty meters outside the building. We believe TrackIO is a promising first step in active localization from outside the building. While TrackIO provides a fully working system where none exists today, we plan to continue to explore avenues to further improve accuracy, resilience, and redundancy.

## References


[1] Adafruit ultimate gps breakout. https://www.adafruit.com/product/746.

[2] ADIB, F., HSU, C.-Y., MAO, H., KATABI, D., AND DURAND, F. Capturing the human figure through a wall. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 219.

[3] ADIB, F., KABELAC, Z., AND KATABI, D. Multi-person localization via rf body reflections. In *NSDI* (2015), pp. 279–292.

[4] BAHL, P., AND PADMANABHAN, V. N. Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* (2000), vol. 2, Ieee, pp. 775–784.

[5] BAHL, P., AND PADMANABHAN, V. N. Radar: An in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE* (2000), vol. 2, Ieee, pp. 775–784.

[6] BENINI, A., MANCINI, A., AND LONGHI, S. An imu/uwb/vision-based extended kalman filter for mini-uav localization in indoor environment using 802.15. 4a wireless sensor network. *Journal of Intelligent & Robotic Systems 70*, 1-4 (2013), 461–476.

[7] BESPOON. Precise location rtls. http://bespoon.com/technology/precise-location-rtls/.

[8] BORRE, K., AKOS, D. M., BERTELSEN, N., RINDER, P., AND JENSEN, S. H. *A software-defined GPS and Galileo receiver: a single-frequency approach*. Springer Science & Business Media, 2007.

[9] CHINTALAPUDI, K., PADMANABHA IYER, A., AND PADMANABHAN, V. N. Indoor localization without the pain. In *Proceedings of the sixteenth annual international conference on Mobile computing and networking* (2010), ACM, pp. 173–184.

[10] DECAWAVE. Decawave. http://www.decawave.com/.

[11] DECAWAVE. DW1000 User Manual. https://decawave.com/content/dw1000-user-manual".

[12] DEVELOPER, D. Mobile sdk. https://developer.dji.com/mobile-sdk/documentation/introduction/index.html.

[13] DJI. DJI Phantom 4 Specs.

[14] DJI. Mavic pro specs. https://www.dji.com/mavic/info#specs.

[15] DUEWEL, E., AND STODDARD, S. After dark, drone is 'best friend a firefighter could have'. https://www.usnews.com/news/best-states/oregon/articles/2018-08-11/after-dark-drone-is-best-friend-a-firefighter-could-have.

[16] FAHY, R. F., LEBLANC, P. R., AND MOLIS, J. L. Firefighter fatalities in the united states - 2017. *NFPA* (2018).

[17] FERNANDEZ-MADRIGAL, J.-A., CRUZ-MARTIN, E., GONZALEZ, J., GALINDO, C., AND BLANCO, J.-L. Application of uwb and gps technologies for vehicle localization in combined indoor-outdoor environments. In *ISSPA* (2007), pp. 1–4.

[18] GOWDA, M., DHEKNE, A., SHEN, S., CHOUDHURY, R. R., YANG, L., GOLWALKAR, S., AND ESSANIAN, A. Bringing iot to sports analytics. In *NSDI* (2017), pp. 499–513.

[19] GOWDA, M., MANWEILER, J., DHEKNE, A., CHOUDHURY, R. R., AND WEISZ, J. D. Tracking drone orientation with multiple gps receivers. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking* (2016), ACM, pp. 280–293.

[20] GUTIERREZ, J. A., CALLAWAY, E. H., AND BARRETT, R. *IEEE 802.15.4 Low-Rate Wireless Personal Area Networks: Enabling Wireless Sensor Networks*. IEEE Standards Office, New York, NY, USA, 2003.

[21] HEDGECOCK, W., MAROTI, M., LEDECZI, A., VOLGYESI, P., AND BANALAGAY, R. Accurate real-time relative localization using single-frequency gps. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems* (2014), ACM, pp. 206–220.

[22] HEDGECOCK, W., MAROTI, M., SALLAI, J., VOLGYESI, P., AND LEDECZI, A. Regtrack: a differential relative gps tracking solution. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services* (2013), ACM, pp. 475–476.

[23] HRABIA, C.-E., HESSLER, A., XU, Y., BREHMER, J., AND ALBAYRAK, S. Efffeu project: Efficient operation of unmanned aerial vehicles for industrial fire fighters. In *Proceedings of the 4th ACM Workshop on Micro Aerial Vehicle Networks, Systems, and Applications* (New York, NY, USA, 2018), DroNet'18, ACM, pp. 33–38.

[24] HUGHES, W. J. T. C. N. T., AND TEAM, E. Global positioning system (gps) standard positioning service (sps) performance analysis report. *WAAS FAA*, 101 (2018).

[25] IBRAHIM, M., LIU, H., JAWAHAR, M., NGUYEN, V., GRUTESER, M., HOWARD, R., YU, B., AND BAI, F. Verification: Accuracy evaluation of wifi fine time measurements on an open platform. In *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking* (2018), ACM, pp. 417–427.

[26] IRISH, A., ISAACS, J., ILAND, D., HESPANHA, J., BELDING, E., AND MADHOW, U. Shadowmaps, the urban phone tracking system. In *Proceedings of the 20th annual international conference on Mobile computing and networking* (2014), ACM, pp. 283–286.

[27] JIMÉNEZ, A. R., AND SECO, F. Comparing decawave and bespoon uwb location systems: Indoor/outdoor performance analysis. In *IPIN* (2016), pp. 1–8.

[28] JOSHI, K. R., BHARADIA, D., KOTARU, M., AND KATTI, S. Wideo: Fine-grained device-free motion tracing using rf backscatter. In *NSDI* (2015), pp. 189–204.

[29] KEMPKE, B., PANNUTO, P., AND DUTTA, P. Polypoint: Guiding indoor quadrotors with ultra-wideband localization. In *Proceedings of the 2nd International Workshop on Hot Topics in Wireless* (2015), ACM, pp. 16–20.

[30] KOK, M., HOL, J. D., AND SCHÖN, T. B. Using inertial sensors for position and orientation estimation. *arXiv preprint arXiv:1704.06053* (2017).

[31] KOTARU, M., JOSHI, K., BHARADIA, D., AND KATTI, S. Spotfi: Decimeter level localization using wifi. In *ACM SIGCOMM Computer Communication Review* (2015), vol. 45, ACM, pp. 269–282.

[32] KULMER, J., HINTEREGGER, S., GROSSWINDHAGER, B., RATH, M., BAKR, M. S., LEITINGER, E., AND WITRISAL, K. Using decawave uwb transceivers for high-accuracy multipath-assisted indoor positioning. In *Communications Workshops (ICC Workshops), 2017 IEEE International Conference on* (2017), IEEE, pp. 1239–1245.

[33] KUMAR, S., GIL, S., KATABI, D., AND RUS, D. Accurate indoor localization with zero start-up cost. In *Proceedings of the 20th annual international conference on Mobile computing and networking* (2014), ACM, pp. 483–494.

[34] MASSATT, P., AND RUDNICK, K. Geometric formulas for dilution of precision calculations. *Navigation 37*, 4 (1990), 379–391.

[35] NI, L., WANG, Y., TANG, H., YIN, Z., AND SHEN, Y. Accurate localization using lte signaling data. In *2017 IEEE International Conference on Computer and Information Technology (CIT)* (Aug 2017), pp. 268–273.

[36] PALACIOS, J., CASARI, P., AND WIDMER, J. Jade: Zero-knowledge device localization and environment mapping for millimeter wave systems. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE* (2017), IEEE, pp. 1–9.

[37] PANNUTO, P., KEMPKE, B., CHUO, L.-X., BLAAUW, D., AND DUTTA, P. Harmonium: Ultra wideband pulse generation with band-stitched recovery for fast, accurate, and robust indoor localization. *ACM Trans. Sen. Netw. 14*, 2 (June 2018), 11:1–11:29.

[38] PARKINSON, B., AND ENGE, P. Differential gps. In *Global Positioning System: Theory and Applications* (1996), American Institute of Aeronautics and Astronautics, pp. 3–50.

[39] PIX4D. Do rtk/ppk drones give you better results than using gcps? https://pix4d.com/rtk-ppk-drones-gcp-comparison/.

[40] PROROK, A., ARFIRE, A., BAHR, A., FARSEROTU, J. R., AND MARTINOLI, A. Indoor navigation research with the khepera iii mobile robot: An experimental baseline with a case-study on ultra-wideband positioning. In *2010 International Conference on Indoor Positioning and Indoor Navigation* (Sept 2010), pp. 1–9.

[41] PROROK, A., TOMÉ, P., AND MARTINOLI, A. Accommodation of nlos for ultra-wideband tdoa localization in single- and multi-robot systems. In *2011 International Conference on Indoor Positioning and Indoor Navigation* (Sept 2011), pp. 1–9.

[42] RAI, A., CHINTALAPUDI, K. K., PADMANABHAN, V. N., AND SEN, R. Zee: Zero-effort crowdsourcing for indoor localization. In *Proceedings of the 18th annual international conference on Mobile computing and networking* (2012), ACM, pp. 293–304.

[43] RUIZ, A. R. J., AND GRANJA, F. S. Comparing ubisense, bespoon, and decawave uwb location systems: Indoor performance analysis. *IEEE Transactions on Instrumentation and Measurement 66*, 8 (Aug 2017), 2106–2117.

[44] RUIZ, A. R. J., GRANJA, F. S., HONORATO, J. C. P., AND ROSAS, J. I. G. Accurate pedestrian indoor navigation by tightly coupling foot-mounted imu and rfid measurements. *IEEE Transactions on Instrumentation and measurement 61*, 1 (2012), 178–189.

[45] SCZYSLO, S., SCHROEDER, J., GALLER, S., AND KAISER, T. Hybrid localization using uwb and inertial sensors. In *2008 IEEE International Conference on Ultra-Wideband* (Sept 2008), vol. 3, pp. 89–92.

[46] SHEN, G., CHEN, Z., ZHANG, P., MOSCIBRODA, T., AND ZHANG, Y. Walkie-markie: Indoor pathway mapping made easy. In *Proceedings of the 10th USENIX conference on Networked Systems Design and Implementation* (2013), USENIX Association, pp. 85–98.

[47] SPIEKER, S., AND ROHRIG, C. Localization of pallets in warehouses using wireless sensor networks. In *2008 16th Mediterranean Conference on Control and Automation* (June 2008), pp. 1833–1838.

[48] UBISENSE. Ubisense solutions. http://www.ubisense.net/ubisense-solutions.

[49] VAN DE VELDE, S., AND STEENDAM, H. Cupid algorithm for cooperative indoor multipath-aided localization. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on* (2012), IEEE, pp. 1–6.

[50] VAN DIGGELEN, F., AND ENGE, P. The worlds first gps mooc and worldwide laboratory using smartphones. In *Proceedings of the 28th International Technical Meeting of The Satellite Division of the Institute of Navigation (ION GNSS+ 2015)* (2015), pp. 361–369.

[51] VASISHT, D., KUMAR, S., AND KATABI, D. Decimeter-level localization with a single wifi access point. In *13th USENIX Symposium on Networked Systems Design and Implementation (NSDI 16)* (2016), pp. 165–178.

[52] WANG, H., SEN, S., ELGOHARY, A., FARID, M., YOUSSEF, M., AND CHOUDHURY, R. R. No need to war-drive: Unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services* (2012), ACM, pp. 197–210.

[53] WANG, H., SEN, S., MARIAKAKIS, A., ROY CHOUDHURY, R., ELGOHARY, A., FARID, M., AND YOUSSEF, M. Unsupervised indoor localization. In *Proceedings of the 10th international conference on Mobile systems, applications, and services* (2012), ACM, pp. 499–500.

[54] XIONG, J., AND JAMIESON, K. Arraytrack: a fine-grained indoor location system. In *Presented as part of the 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)* (2013), pp. 71–84.

[55] YANG, Z., WU, C., AND LIU, Y. Locating in fingerprint space: wireless indoor localization with little human intervention. In *Proceedings of the 18th annual international conference on Mobile computing and networking* (2012), ACM, pp. 269–280.

[56] YARLAGADDA, R., ALI, I., AL-DHAHIR, N., AND HERSHEY, J. Gps gdop metric. *IEE Proceedings-radar, sonar and navigation 147*, 5 (2000), 259–264.

[57] YOUSSEF, M., AND AGRAWALA, A. The horus wlan location determination system. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services* (2005), ACM, pp. 205–218.