Experiences with Performance Tradeoffs in Practical, Continuous Indoor Localization

Azeem J. Khan
Oriental Institute of Management
University of Mumbai
azeem@oes.ac.in

Vikash Ranjan
Senior Engineer
RedMart Pte Ltd.
vikash@redmart.com

Trung-Tuan Luong and Rajesh Balan and Archan Misra
School of Information Systems
Singapore Management University
{ttluong,rajesh,archanm}@smu.edu.sg

***Abstract*—This paper describes our experiences and observations with a localization system that continuously tracks the indoor location of a large number of consumer mobile devices. Unlike past work that focuses principally on the accuracy of the location tracking algorithm, we study the performance of the localization system in terms of key additional metrics: scalability and energy-efficiency, which can sometimes conflict with the desire for high accuracy. To ensure that our solution can handle both Android and iOS-based mobile devices (& other closed mobile platforms), we adapt the conventional *client-side* fingerprinting-based localization approaches to develop a novel and practical *infrastructure-based* location tracking strategy. We study the relative accuracy to the two approaches in two different types of indoor buildings. Our studies establish how the building and its occupancy characteristics affect the accuracy achievable by different algorithms, and provide insights into why scalable, energy efficient and accurate indoor location tracking remains a challenge in practice.**

***Keywords*-indoor localization; urban sensing;**

I. Introduction

As part of the LiveLabs project [1] at Singapore Management University (SMU), we are currently building an urban-scale testbed that (a) permits real-time gathering of fine-grained context data from over 30,000 participants in multiple urban spaces (mostly indoors), and (b) enables the testing of next-generation mobile services and applications that utilize such context. As location remains one of the most important participant contexts, we are developing and evaluating solutions that can *continuously* track the indoor location of a large participant population.

Most indoor localization techniques, in our view, have focused almost exclusively on *accuracy*, demonstrated through limited experiments performed on a small set of devices. In addition, most of the traditional Wi-Fi fingerprinting based approaches implicitly assume that the mobile clients are able to periodically query the signal strengths of all the Wi-Fi APs deployed in the indoor environment. However, both of these attributes need to be re-examined when building a solution that needs to scale to numerous commodity smartphones. In particular:

- Between the two most dominant smartphone operating systems, only the Android OS offers an open API to retrieve Wi-Fi scan data; iOS based devices (which comprise approx. 70% of the Singaporean smartphone market) offer no such functionality. Hence, our solution must also work for smartphones that do not allow client-side measurement of Wi-Fi AP RSSI values.
- For a large-scale, continuous location tracking framework, we must carefully consider two additional performance metrics: *scalability* and *energy-efficiency*. Clearly, a practical solution must permit the simultaneous tracking of 100s to 1000s of collocated mobile devices, and must ensure that the battery drain is not severe enough to hamper the lifestyles of consumers.

This paper describes our recent experiences, in these two important systems areas, in building and evaluating the first version of such a continuous location tracking system in real life, *public* indoor spaces. More specifically, we have built indoor location tracking software for both the Android and iOS platforms and shall present our experimental insights obtained by testing this software in two public indoor locations:

1) **Mall**: a large, and very crowded shopping mall in central Singapore.
2) **SIS**: School of Information System campus building at SMU.

We emphasize at the outset that our solutions are not research prototypes—they work with the pre-existing commercial Wi-Fi enterprise infrastructure already deployed in these locations. We shall not only discuss how the features/limitations of current commercial enterprise Wi-Fi solutions affect the metrics of *accuracy* and *scalability*, but also scrutinize how a variety of differences in the two environments (such as the number/density of deployed APs, the building layout and time-varying changes in the density of inhabitants) impact our performance metrics.

Research Questions: Our main research questions are the following:

- How does one design a "server-based" continuous

This work was performed at Singapore Management University by Azeem J. Khan while he was a Research Fellow and by Vikash Ranjan while he was a Research Engineer

[0]This research is supported by the National Research Foundation Singapore under its IDM Futures Funding Initiative, and administered by the Interactive & Digital Media Program Office, Media Development Authority.

location tracking solution (for iOS devices), and how does the resulting location tracking accuracy presently compare with more conventional client-side fingerprint-based localization (performed on Android devices)?
- What characteristics of the indoor environment (such as the number/density of distinct APs, the mix of pressurized-indoor and non-pressurized outdoor spaces, and the occupant density) affect the location system's accuracy, and by how much?
- What are the performance limitations or characteristics exhibited by representative commercial Wi-Fi infrastructures (APs & controllers) today, and how do such infrastructure characteristics lead to possible tradeoffs between *scalability* vs. *accuracy* for practical infrastructure-based location tracking?
- How do the relative energy-overheads of different sensors (specifically, Wi-Fi scanning, accelerometer & compass), when used for continuous location sensing, affect the *energy-efficiency* vs. *accuracy* tradeoffs for both Android and iOS devices?

Key Contributions: We emphasize that our focus is not on designing theoretically-new localization algorithms, but instead on understanding the practical, systems-related issues that impact the performance of a "representative" set of algorithms. Our main objective is to gain a better understanding of how localization algorithms fare in the real world. Accordingly, we make the following key and useful contributions in this paper:

1) **Describe localization algorithms for Android and iOS and compare them.** We describe a relatively straightforward localization strategy that combines Wi-Fi fingerprinting with inertial motion estimation via a Viterbi-based algorithm for temporal location tracking. While the implementation of this generic approach for Android devices is straightforward, we propose a novel Wi-Fi controller-assisted "limited-reverse-fingerprinting" based mechanism solution for iOS devices. We then compare the relative accuracy achievable on the Android and iOS devices.
2) **Study sensitivity of results to variations in building characteristics and occupancy:** We show that the accuracy results achieved are independent of some of the algorithm parameters (e.g., depth of the Viterbi tree), but are quite susceptible to building-specific variations, some of which are static (such as the number of distinct APs) and some of which are quite dynamic (such as the occupant density). These observations help establish the benefits of real-time RF fingerprints, which can help deal with time-varying changes in the indoor radio environment.
3) **Reveal the performance limitations of existing controller-based solutions:** We demonstrate that Wi-Fi infrastructure-centric solutions, which rely on the ability to continuously query either the Wi-Fi controller (or individual APs) to dynamically retrieve "reverse signal strength" information, can suffer due to possible performance bottlenecks in commercial Wi-Fi controllers and the staleness of readings reported by the APs, and then demonstrate how this bottleneck leads to a tradeoff between individual device location accuracy and overall system scalability (number of devices simultaneously tracked).
4) **Identify the necessity of manual tuning** Our experience illustrates what we believe to be an important, often-overlooked drawback of current localization technology. We discover that the location tracking performance is quite sensitive to the specific properties of each individual indoor space, and its usage, implying that the location tracking accuracy cannot be universally predicted, but shows site-specific variations and requires customized adjustments.

	Mall	**SMU**
Number of Floors	7	5
Indoor/Outdoor	Fully-Indoor	Mixed Indoor+ Outdoor (Floor 1&2)
Avg. Floor Area (sq.m)	5000[1]	3000
Avg. Store/ Room Width (m)	8	3
No. of Wi-Fi APs/floor	7	12
Avg. AP Distance (m)	15	8
No. of Fingerprint Landmarks	26 (floor 1) 27 (floor 2)	67 (floor 2) 76 (floor 4)

Table I
MALL STATISTICS

II. BUILDINGS AND THE DATA COLLECTION PROCESS

To evaluate our developed large-scale Wi-Fi based indoor location tracking system, we have focused on two different 'public' buildings (both of which are key LiveLabs testbed venues):

- **Mall**: The first building is a very large (over 800,000 sq. ft of indoor space) shopping mall near our University with multiple floors and a 'well-shaped' internal structure: all floors above the first floor can look onto the first floor. The building is rectangular in shape. The entire building is closed and has a central HVAC. For the purposes of this study, we concentrated on evaluating our location tracking technologies on two representative floors: 1 & 2 (which are also the two busiest floors in the mall).
- **SIS**: The second building is the School of Information Systems (SIS) building in our University. SIS is a 5 storey, almost L-shaped structure, with the shorter of the two segments having a partially circular boundary at the extremity. The first floor is open (non-pressurized), containing the visitor waiting and security/reception areas. The second floor is also partially open: while the

[1]Of the total floor area of $\sim 10,000$ sq.m, only about 5000 sq.m., was directly accessible for consumers.

seating area and corridors are non-pressurized, the class rooms, the meeting rooms and the research centres are closed and controlled by the central HVAC. The top 3 floors are all pressurized and centrally air-conditioned.

Table I details some of the key attributes of these two buildings, including the size and the nature of Wi-Fi AP deployment. It is important to point out that both buildings exhibit variation in the occupant density, but at different scales. More specifically, *Mall* shows very high visitor loads in the evenings and weekends (and is relatively less congested during the daytime on weekdays). While such variations are less pronounced at SIS, we observe higher densities (especially on floors 1-3) by students during class hours. The labs and staff offices, located on floors 4 & 5, see relatively less fluctuation in occupancy loads.

A. The Data Collection Process

Our location tracking solutions are based on *fingerprinting*-namely, the (for now, offline) collection of RF measurements at known 'landmark' points within the buildings (see Table I for details). To generate the RF fingerprint maps, we employed two slightly different strategies for iOS and Android devices. For Android devices, we use a custom application that scanned for Wi-Fi access points across all channels once every 50ms. The application collected the ⟨*timestamp, RSSI, AP BSSID*⟩ of each AP for these scans for at least 2500 seconds. This was performed at 53 different landmark locations in Mall (26 on floor 1 and 27 on floor 2) and 143 locations in SIS, and was repeated 6 times, across different days and at different times of the day. These measurements were used to construct fingerprint maps for each building.

For the iOS platform (where the network level information such as BSSID of APs and RSSI measurements are not available on the mobile device), we employed a "reverse" fingerprinting strategy, with the iPhone's uplink SNR value being logged and measured at the AP. Due to various constraints we faced with the deployed Wi-Fi infrastructure (to be discussed in Section IV), we had to perform SNR sampling at much lower frequency (1 Hz); at each landmark, we gathered at least 300 samples. This data was collected at 3 different times of the day on 2 different days, for a total 6 sets of readings.

In all cases, the data was collected with the user standing stationary (facing each of the 4 cardinal directions) at each landmark. To capture the directionality of movement, the user took both a clockwise and counter-clockwise walk along the path connecting the landmarks. The offline fingerprinting and online performance evaluation were performed using the Samsung Galaxy S3 phones (for the Android platform), and the iPhone 4 (for the iOS platform).

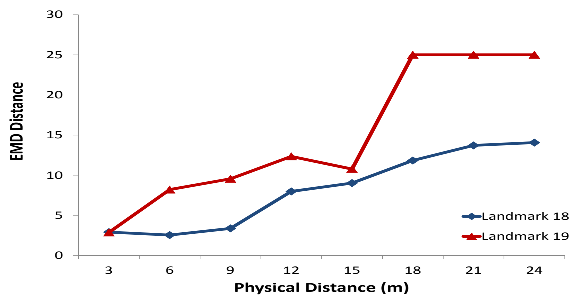
B. Choosing the Landmark Granularity

Any fingerprinting-based algorithm must first define the spatial granularity of the fingerprinted landmarks—in effect, this decides the lower bound (best case) of the location tracking error. We first measured the Wi-Fi signal strength variation in the two buildings to ascertain the useful spatial granularity for landmarks–i.e., the minimum change in location that resulted in a perceptible change in the RF measurements.

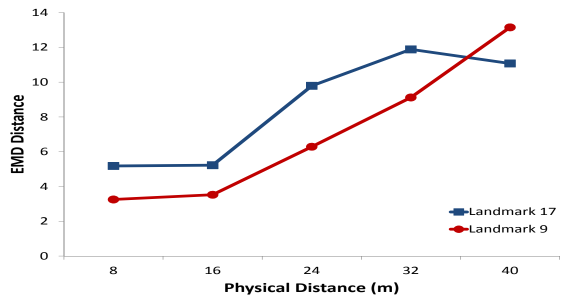
The two buildings have very different layouts, which impact the spatial variation of RF signals. $Mall$ has a rectangular shape and a symmetric, "open" layout–all the shops are located along the periphery of the rectangle, with almost no obstructions or walks in the interior. The SIS building, on the other hand, is maze-like, with many sharp corners, interior rooms and associated walls. Consequently, in SIS, even small position changes can alter the line-of-sight to certain APs, and thus lead to significant changes in RF signal strengths.

To understand these effects, we measured signal-strength in $Mall$ at a granularity of 4 meters (with stores typically being 8 meters wide, our landmarks are effectively half-a-store apart). In SIS, we used the regularly-spaced fire sprinkler points, spaced 3 meters apart, as our initial landmarks. Each fingerprinted landmark is represented by a L dimension vector $[(AP_i, SignatureAP_i)]$. The 'signature' of AP depends on the specific Wi-Fi localization algorithm employed: it is the mean of signal strength for RADAR and the probability mass function (p.m.f.) for HORUS [2], another Wi-Fi localization technique. To understand the spatial sensitivity of these fingerprints, we used the more detailed p.m.f. (as this provides more discriminatory power than simple averages), and computed the Earth Mover Distance (EMD) [3] (a measure of dissimilarity between two probability distributions) between a selected landmark and its nearby points. As shown in Fig. 1, our plots show the EMD of 2 selected landmarks, separately for SIS (landmarks 18 and 19 on level 4) of SIS and $Mall$ (landmark 9 and 17 on level 2). In level 4 of SIS, landmark 18 lies along a long corridor, while landmark 19 lies near the intersection of two paths. As a consequence, as seen in Figure 1a., the EMD plots show that the RSSI measurements begin to diverge only after $\approx$ 9 meters at landmark 18, while they exhibit a much higher spatial sensitivity for landmark 19. In $Mall$ (see Figure 1b.) on the other hand, the EMD metric remains relatively unchanged until we move at least 16 meters away (roughly corresponding to 2 store widths).

Key Insight: Our measurements clearly show that Wi-Fi RSSI values have different spatial sensitivity in different buildings. In SIS, the RF measurements show meaningful divergence after a separation of $\approx$ 6 meters, whereas the divergence becomes meaningful only at about 16 meter separation in the 'open-layout' $Mall$. These results suggest that the location tracking accuracy of purely Wi-Fi fingerprinting-based techniques can, at best, be about 6 meters in SIS and about 16 meters ($\pm$1 store) in $Mall$. The performance of Wi-Fi based localization algorithms is



a. **SIS**



b. **Mall**

Figure 1. Earth Mover Distance (EMD) Metric as a function of distance from reference landmark

thus *not universal*, and will vary appreciably depending on the building structure and layout. *In particular, fine-grained indoor localization is particularly challenging in more 'open' structures, commonly found in malls.*

III. The Base Localization Algorithms

In this section, we describe our basic algorithms for indoor localization software, first focusing on the Android implementation (which permits scanning and real-time recording of Wi-Fi signal strengths on the phone), and then discussing the corresponding analogue for iOS devices. (Modifications specifically tailored to address limitations imposed by an infrastructure-driven solution for the iOS platform will be discussed separately in Section IV.)

Our base localization strategy employs a two-step approach, conceptually illustrated in Figure 2. The two-step approach seeks to combine the benefits of Wi-Fi fingerprinting (which defined the first generation of indoor location approaches) with the power of sensor-based motion estimation (made possible by the recent availability of multiple sensors, such as the accelerometer and compass, on commercial smartphones) and involves the following steps:

1) In the first step, a Wi-Fi fingerprint-based location technique *periodically* (approximately every few seconds) produces a list of estimated positions (at successive time instants) ordered in decreasing order of probabilities.
2) In a logically separate process, an inertial sensor-sampling module computes a set of possible motion vectors (i.e. the angular direction and movement distance) taken by the user *within* these time instants.
3) Finally, we employ a Viterbi-like *path likelihood estimation* algorithm (described in Section III-C) that combines the outputs of the Wi-Fi location estimator and the inertial motion estimator to determine the most likely path (i.e., *temporal* sequence of locations visited), and hence obtain the most likely location.

Conceptually, the use of motion vectors and the Viterbi-like path likelihood computation process should enable us to

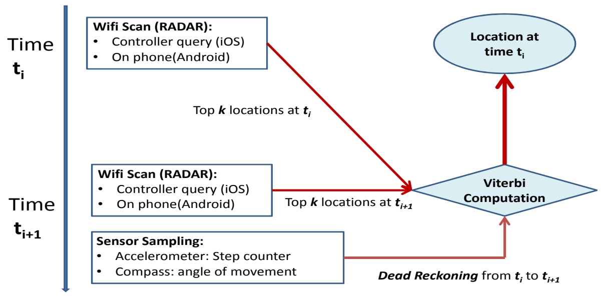


Figure 2. The location estimation algorithm

"smoothen" the estimated output, avoiding certain observed jumps or teleportation effects between successive Wi-Fi based location estimates, and helping track the evolution of location between 'landmarks'. However, from a systems perspective, this potential additional accuracy comes with extra 'energy cost' (the sensing on the device's sensors)—a tradeoff that we shall revisit in Section V-B.

This conceptual and modular architecture allows us to independently experiment with different algorithms for each of the steps above. For example, in this paper, we will report on the use of RADAR as an exemplar of the Wi-Fi location estimator. (We have implemented a HORUS-based solution as well, but do not discuss details of those results here as they are overall very similar to RADAR.) Likewise, in this paper, we shall use an accelerometer-based step-counting and compass-based angular direction computation strategy, but these may easily be substituted in future versions by more advanced motion estimators (e.g., additionally using the gyroscope). Similarly, different variants of the base Viterbi-like path estimation algorithm may be used as well.

A. *RADAR implementation for Android and iPhone*

We first describe a RADAR-based implementation of the Wi-Fi fingerprinting-based location estimation algorithm. This algorithm is conceptually applicable to both Android and iPhone devices, with the distinction that the iPhone version requires 'reverse fingerprinting', i.e., an offline training phase that records the phone's signal strength (as measured by its associated AP) at known 'landmark' points. The offline training process for RADAR is straightforward.

For the Android implementation, the phone performs a Wi-Fi scan at each *landmark* location. For each landmark, it then computes the $mean$ of the RSSI values (independently for each of the APs observed during the scan), effectively creating a M dimensional vector (where M is the total number of APs in that indoor facility). Finally, we create a fingerprint map (of dimension {number of APs}x{number of landmarks}) consisting of multiple tuples of the form computed for each landmark location L_i as follows:

$$\begin{pmatrix} \dots & \\ L_i, & [\overline{RSSI^i_{AP_1}}, \dots, \overline{RSSI^i_{AP_M}}] \\ L_{i+1}, & [\overline{RSSI^{i+1}_{AP_1}}, \dots, \overline{RSSI^{i+1}_{AP_M}}] \\ \dots & \end{pmatrix}$$

In the online phase, the user's (Android) smartphone performs multiple scans of the Wi-Fi access points (typical scanning frequency is 20 Hz), at a given time t, and then computes the mean for the sampled RSSI values of each AP, which is presented as following m(t) vector:

$$m(t) = [\overline{RSSI^*_{AP_1}}, \dots, \overline{RSSI^*_{AP_M}}]$$

The RADAR location estimation algorithm then calculates the Euclidean distance (in the M-dimensional RF space) of these mean RF RSSI vector $m(t)$ from the M-dimensional points (associated with the landmark positions) in the RF fingerprint map created earlier. Our implement then picks the top K nearest landmarks as those with the smallest Euclidean distance values, and then assigns each such landmark a *probability* that is inversely proportional to the corresponding Euclidean distance.

For the iPhone, the mathematical steps, of the RADAR-based computation of the most-probable locations, remains fundamentally unchanged. However, as we shall see later (in Section IV-B), the iPhone fingerprint will consist of the single AP to which the phone is *associated* and the SNR value (of the phone, as measured by that AP). Accordingly, in this case, at any given landmark, the fingerprint map will simply consist of a tuple: $\langle L_i, AP_i, \overline{SNR_{AP_i}} \rangle$, where AP_i denotes the AP with which the phone associates while stationary at landmark L_i. Neither the fingerprint nor the actual 'scan' can now use the signal strength values measured by the other APs. In this case, the $top - K$ values returned are restricted solely to that set of locations that are within the 'association zone' of the current AP, and the M-dimensional RF vector is now replaced by a 1-dimensional SNR value.

B. Inertial motion computation

The inertial motion vector is computed *continuously*, in-between those instants when the Wi-Fi fingerprint-based likely locations are estimated. We utilize the two embedded sensors, accelerometer and compass, for computing the

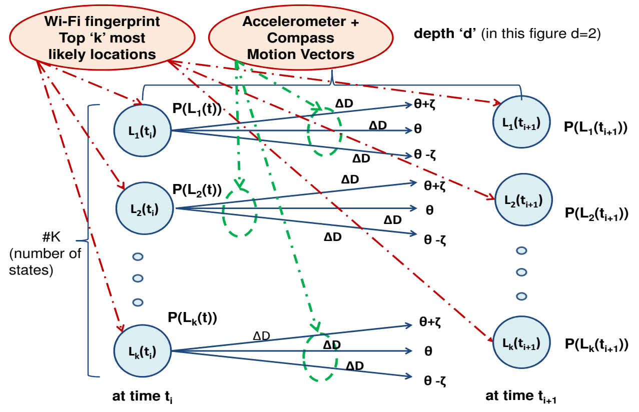


Figure 3. The Viterbi algorithm used to estimate current location.

distance and the angular movement direction respectively. (As both the Android and iOS APIs allow for the retrieval and use of these two sensors' values, this algorithm is effectively identical on both platforms.) The accelerometer readings (sampled at 50 Hz) are first transformed into a 'step-count' estimate, which is then multiplied by a tread length to obtain the effective movement distance (denoted by ΔD). To accommodate diversities across different individuals, without requiring individual-specific training and customization, we simply use 3 separate tread length values $\{0.45m, 0.5m, 0.55m\}$, thereby creating 3 different estimates for ΔD.

The magnetic compass readings are integrated (with ΔD) to obtain the net displacement vector of the user. Unfortunately, magnetic compasses are known to suffer from various indoor artefacts (e.g., readings fluctuate significantly near elevators and escalators due to the presence of ferromagnetic material). We empirically observed that, even at rest on our office table for 20 minutes, the phones' compass readings fluctuate with a Gaussian distribution, with 90% of the readings lying within a range of ± 1 degree. Accordingly, we also take three possible compass readings into our computation—the mean value θ, $\theta - 1$ and $\theta + 1$. By allowing 3 possible values for ΔD, and three possible values for the angular direction this module effectively generates 9 equi-probable displacement vector estimates.

C. Viterbi-like path estimation algorithm

Our Viterbi-like algorithm probabilistically combines the periodic Wi-Fi location estimates with the intervening motion vector estimates to compute the relative likelihood of different 'motion paths' (evolution of the location over time), and thus helps select the most likely location (taking into account the temporal correlation and activity behavior of each individual). The basic principles of this algorithm (adapted from [4]) are illustrated in Figure 3. The Viterbi algorithm is associated with a parameter called $depth$, where $depth$ effectively denotes the number of consecutive time instants

Algorithm 1 Indoor Localization

```
1: The objective of the algorithm is to detect the best location at time $t$ in the past when user is at time $t+1$
2: procedure RADAR PROBABILITY
3:     Calculate the best-k locations at times $t_i$ and $t_{i+1}$
4:     $P(L_j(t_i));\ j = 1, \ldots, K$
5:     $P(L_j(t_{i+1}));\ j = 1, \ldots, K$
6: end procedure
7: procedure DEAD-RECKONING PROBABILITY
8:     Calculate the conditional probability of connecting $L_m(t_i) \to L_n(t_{i+1})$
9:     Dead-reckoning provides set S of 9 expected locations of $L_m(t_i)$ at time $t_{i+1}$: $L^j_m(t_i);\ j = 1, \ldots, 9$
10:    Among 9 locations in S, $L^n_m(t_i)$ denotes the closest point to $L_n(t_{i+1})$
11:    $P(L_n(t_{i+1})|L_m(t_i)) = \frac{\frac{1}{d(L^n_m(t_i), L_n(t_{i+1}))^2}}{\sum_{j=1}^{k} d(L^j_m(t_i), L_j(t_{i+1}))^2}$
12: end procedure
13: procedure BEST-PATH SELECTION
14:    Calculate the probability of path $L_m(t_i) \to L_n(t_{i+1})$ occurs
15:    $P(L_m(t_i) \to L_n(t_{i+1})) = P(L_m(t_i)) * P(L_n(t_{i+1})|L_m(t_i)) * P(L_n(t_{i+1}))$
16:    There are $k^2$ paths, path with highest probability is selected $\to$ identify the location at time t
17: end procedure
```

that are used to find the *path probability* i.e. likelihood of a path being taken, for each individual. Algorithm 1 outlines the steps in our Viterbi computation for $depth = 2$. Note that, due to the nature of the algorithm, the computed location is *always one of the 'landmarks' established earlier* (as the choices for $L(t_i)$ and $L(t_{i+1})$ are always limited to these fingerprint landmarks).

IV. iPhone and other closed systems

As mentioned before, the iOS platform (and other closed platforms such as Windows Mobile and Blackberry) do not allow a user-space application on the mobile device to receive signal strength estimates of all the nearby Wi-Fi APs (or even that of the currently-associated AP), making the use of traditional client-side fingerprinting-based localization techniques impossible. We have thus investigated and implemented a 'server-side' solution, where we interface with one or more commercially-deployed Wi-Fi "controllers" (which control the operation of multiple APs) to retrieve the signal strength information on the uplink (between the mobile device and individual APs).

On investigating the properties of the enterprise-grade Wi-Fi networks deployed at SMU and the *Mall*, we found that each AP kept track of the signal strength (reported as signal-to-noise ratio or **SNR**) of mobile devices that were actively *associated* with it. The APs did not provide any information about clients that were associated with a different AP.

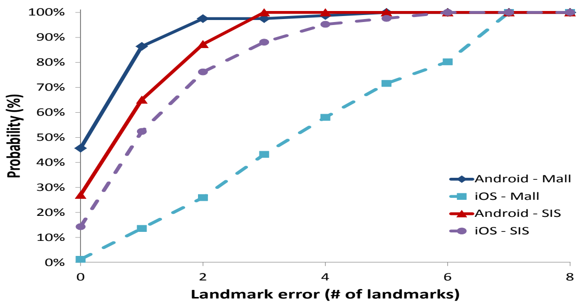


Figure 4. iOS vs Android: indoor localization error. CDF of location error plotted in units of 'landmarks'. The inter-landmark distance is approx. 3m for SIS and 8m for $Mall$.

Accordingly, our infrastructure-assisted approach uses only the knowledge of {*currently associated AP, SNR*} of each mobile device. Note that, although our focus is on the iOS platform, this approach works seamlessly for any mobile device, *independent of its specific OS*.

A. iOS vs Android: indoor localization accuracy

We now investigate the location accuracy when the Wi-Fi fingerprints are based purely on {*currently associated AP, SNR*} values, reported by the Wi-Fi controllers. To implement this infrastructure-assisted approach, the location tracking software now needs both a client and a server component. In our implementation, the iOS device has an application installed, that triggers a query for the associated AP and SNR values to a location server (a Linux PC), which in turn queries the controller with the MAC address of the phone. (We discuss the mechanics of querying the controller in Section IV-B). The server then replies with the SNR value seen by the AP to which the mobile is currently associated. This value is then used for looking up the fingerprint map to produce a list of most-likely k locations, as described in Section III, which is then combined (on the phone) with the steps of inertial motion estimation and Viterbi smoothing (which remain identical to that described in Section III).

For iOS devices, the fingerprint map at each landmark contains the AP's BSSID and the average SNR value seen by that AP when the mobile device was located at that landmark. Due to time-varying changes in the RF environment, as well the hysteresis in AP handoff algorithms, we observed that, depending on direction of movement, *at any landmark location the mobile device can be associated with different APs (at different times).* Accordingly, the fingerprint map at each landmark contains *multiple* tuples of $< connectedAP, SNR >$.

Figure 4 shows the CDF of the location error observed by our client-based implementation (for Android) and the server-side implementation (for iOS), for both *SIS* (level 2) and *Mall* (level 2). It is easy to see that the richer set of APs observed during client-based Wi-Fi scanning allows the Android-based location tracking software to perform much better than the infrastructure-based mechanism used for iOS devices. The Android system could identify the user's location with error of 1 landmark distance at 87% and 65% of the time in $Mall$ and SIS respectively, and was 100% accurate at an error tolerance of ±3 landmarks. In contrast, the iOS solution, based on querying the controller, results in a location error of ±2 landmarks approximately 70% of the time in SIS, and only 25% in the $Mall$. (As discussed shortly, the controller-based solution also suffers from *stale* SNR values provided by the APs.) Here, we made two interesting observations:

1) For the client-side Wi-Fi scanning (on Android), we see that the accuracy (in terms of landmarks) is worse in SIS than in $Mall$. In contrast to $Mall$, SIS has a much denser deployment of APs, leading to a very large dimensionality of the APs in the fingerprint map. The Euclidean distance calculation in RADAR is not robust at handling a very large vector dimensionality (large number of APs). Accordingly, for client-side Wi-Fi fingerprinting based approaches, we suggest restricting the map to *a pre-filtered smaller subset of APs* (e.g., the top-N APs with the highest RSSI values) at each landmark.
2) Conversely, the controller-querying based location tracking for iOS performs much better in SIS than in $Mall$. For the commercial Wi-Fi infrastructures we experimented with, the SNR values are updated only periodically (often once every 3-4 mins!!) while the mobile continues its association with its current AP; a re-association, however, causes an immediate *triggered* update of the SNR values. Due to the denser AP deployment, mobile users perform more frequent inter-AP handoffs in SI, leading to the use of more up-to-date SNR information in location tracking.

B. Controller Limitations

Our server-side implementation of location tracking requires the periodic querying of the Wi-Fi controller(s) for the current SNR value of mobile nodes. Unfortunately, the present generation of Wi-Fi controllers are designed primarily for static management of mobile devices (e.g., authentication setup), and not for supporting the high-throughput retrieval of real time network statistics. The first problem is that the controller receives updated SNR values only when the mobile actually transmits; consequently, the information available on the controller is often very stale. For a non-active mobile, we found the value of the last received transmission varied from 151–209 seconds; this was a contributory factor to the poor performance of the infrastructure-based approach for iOS (Figure 4).

A bigger potential issue is the throughput bottleneck for query when the number of mobiles being monitored is large. We found that the SNR information could be obtained from the controller either via a console-based command line interface (CLI) or via SNMP, both have performance limitations. For the CLI mode, even with automated tools (such

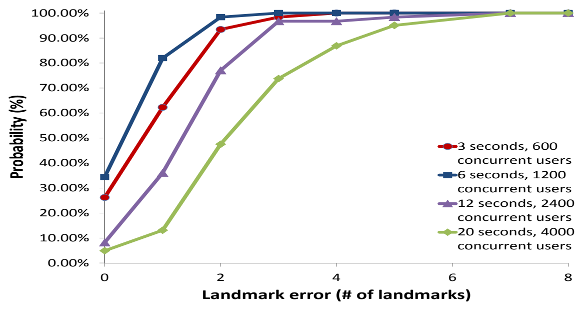


Figure 5. Accuracy of localization vs. Wi-Fi SNR polling interval

as Expect) we were able to obtain a maximum sustained throughput of only 15 queries/sec on the Aruba^{TM}3600 controller. Given that the SNR values for different mobile devices must be obtained individually, this suggests that the *interval between successive SNR queries for a specific device* can grow significantly, when the number of devices being concurrently located increases (e.g., 150 users implies an interval of 10 secs). Conversely, the SNMP-based query throughput is much higher. In our experiments, a controller managing $\approx$60 APs (in our SIS building) and more than a 1000 concurrent users could support a query rate of $\approx$200 records/sec. The drawback, is that the SNMP-based query does not permit the querying for individual mobile nodes; instead, the MIB table contains entries listed by the MAC addresses of APs. Accordingly, to query a single mobile device, the *entire SNMP table must be retrieved and searched.* For example, if we assume 5000 concurrent users in $Mall$, the SNMP-based approach would require a minimal interval (5000/200=) 25 secs between successive SNR queries for the same mobile device.

The above examples illustrate that *there is clearly a dependency between the number of mobile nodes being tracked* and the interval at which localization can be performed for an individual device. This can significantly affect the resulting location accuracy. To establish this impact, we varied the interval between successive Wi-Fi polls performed on the Android device, to *mimic* the impact that longer intervals would have in an infrastructure-based solution. (We avoided running this experiment directly on the controllers, as the IT departments were reluctant to let us stress-test the live Wi-Fi networks.) Figure 5 shows the resulting CDF of the location error in SIS, under different values for the Wi-Fi polling interval (the number of concurrent users that would result in the corresponding interval, under the SNMP query approach, are also provided). It is clear that, as the interval increases (beyond 6 secs), the location accuracy degrades sharply–at an interval of 20 secs, we can achieve an accuracy of ±1 landmark only 10% of the time. Our result clearly establishes an important characteristics of infrastructure-based localization schemes: *there is a tradeoff between individual location accuracy and the number of mobile devices being tracked.* In high-traffic indoor envi-

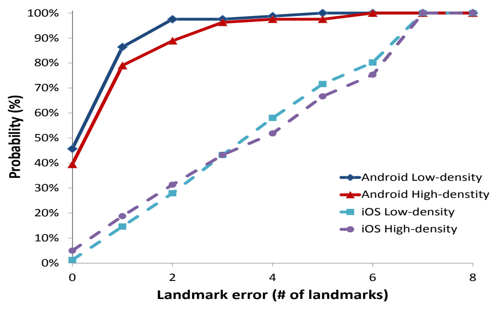


Figure 7. Impact of occupant density on location accuracy in $Mall$

ronments ($Mall$ often has 5000-10,000 concurrent visitors on weekday evenings), this can be a serious limitation for large-scale, continuous location tracking.

V. Issues in a practical deployment

A. Density of people

1) Impact on fingerprint data: While analysing the fingerprint data, we observe that within similar time of the day at the same location, the RSSI seen by the mobile is affected by the density of people in the building. As we used RADAR (using mean of signal strength) and HORUS (using distribution of signal strength) as exemplars of Wi-Fi location estimation, we measure the impact of occupant densities on the RSSI variation ranges, as well as on the mean RSSI value of all the APs.

We measure the variation range of RSSI of all the APs, and calculate the percentage of these values belong to 5 bins: less than 5, 5–10, 10–20, 20–30 and $>$ 30 dBm. Figure 6a. and 6.b. show a common trend for both $Mall$ and SIS buildings, in that the RSSIs have higher variation ranges when the density of people increases. For example, in SIS at low occupant density (2pm) , 44% and 30% of the APs have the RSSI variations belong to 5–10 and 10–20 dBm respectively; however, at high occupant density (7pm), it is 14% and 56% respectively. This clearly shows that about 25% of the APs have shifted from small RSSI variations to large RSSI variations. These observations suggest that we should have different fingerprint maps at different occupant density, especially if we use RSSI *distribution-based* location estimators (such as HORUS).

Figure 6.c and 6.d plot the graphs showing the percentage of APs having the mean RSSI value seen by mobile device fall into 5 bins: < -90, -90 to -80, -80 to -70, -70 to -60, and -60 to -50 dBm (we do not have APs with RSSI value greater than -50dBm). The figures do not show a significant change in the distribution of mean RSSI values (when the mean for each landmark is computed using a longer measurement duration of 6 mins) when there is a change in occupant density.

2) Impact on indoor localization performance: Figure 7 shows the indoor localization performance (using RADAR) with both iOS and Android systems in $Mall$ at different occupant densities (low density at 2pm & high density at 7pm). We have previously seen that higher occupant density implies a wider range of RSSI signals. Accordingly, during indoor location tracking, when the average RSSI values are computed using Wi-Fi scans over short time intervals (2.5 secs), the variation in these average values is likely to be larger when the occupant density is high. Consequently, the location tracking accuracy will be lower when the location estimates are computed using a static fingerprint map.

	WI-FI	WI-FI + Viterbi
Power Consumption (mW)	14.538	251.842
Accuracy $\pm$ 1 landmark	77%	87 %

Table II
Energy consumed vs. Accuracy obtained by different algorithms

B. Energy versus Accuracy

We now study the energy overheads of our location tracking solution. As the SNR value has to be fetched from a controller for closed system, it is plausible to run the entire localization algorithm on a server which can access the controller. But one of the inputs to the Viterbi algorithm is the set of motion vectors, which should be computed on the phone as it requires access to the phone-embedded sensor data. We measured the power drawn for the Wi-Fi and inertial sensing components on a Samsung SII phone over a 20 minute interval; this experiment was repeated 5 times. Table II presents the mean energy usage of these two components (averaged over the 5 repetitions). We observe that the inertial sensors (i.e. accelerometer and magnetometer) dominate the energy usage in the entire indoor localization algorithm, draining an additional 237mW from the battery. Moreover, based on our empirical studies, we note that the use of the Viterbi algorithm (i.e. using inertial sensors) results in a location error within ± 1 landmark 87% of the time; in contrast, the use of Wi-Fi fingerprinting alone provides the same location accuracy only 77% of the time. Clearly, the additional improvement in location accuracy offered by the use of inertial sensing may be attractive for intermittent sensing, but can pose an unacceptable energy overhead when executed *continuously*.

VI. Related Work

Indoor localization has been extensively studied in the research community over the past decade, with the dominant approaches employing various combinations of RF (Wi-Fi) signal strength fingerprinting [5], [2], trilateration and triangulation techniques using RF [6], [7], [8], [9]. Several papers have also studied dead-reckoning techniques using sensors such as accelerometers and compass [10], [11], [12], [13]. Other non-RF based ambient fingerprinting techniques have used sound and light [14], [15], [16]. A fairly comprehensive study [17] of different research proposals noted that no solution fits all environments and that a number of challenges have yet to be addressed.

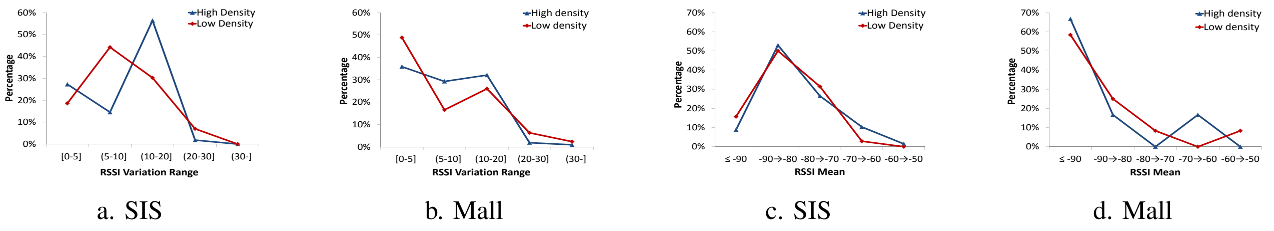


Figure 6. Impact of people density on RSSI variation range and RSSI mean

As far as we know, no proposal so far has looked into addressing the limitations of iOs and other closed mobile platforms. We also focus specifically on the challenge of tracking individuals in indoor locations continuously, and under widely varying occupant densities.

VII. Summary and Future Work

In this paper, we have focused on building a continuous indoor location tracking system that scales both in the number of participants tracked and in the types of mobile OS platforms supported (specifically, Android and iOS). Most prior work on Wi-Fi fingerprinting conveniently assumes that the mobile device can retrieve Wi-Fi AP signal strength readings. Since that is not practical in many devices having significant consumer market share, we have implemented a modified infrastructure-assisted solution that employs 're-verse fingerprinting' by querying the commercial Wi-Fi controller infrastructure; this is supplemented by inertial motion estimates derived from smartphone-embedded sensors. Empirical testing on both a campus building and a busy, large shopping mall show that the localization accuracy not only depends on the building layout, but also on the density of the deployed APs. Our experimental studies reveal two major, but somewhat unappreciated, issues that need to be addressed before a universally-applicable location solution can be implemented: *a)* limitations on the query throughput of commercial controllers lead to an undesirable tradeoff between the number of concurrent devices being located and the localization accuracy, and *b)* time-varying changes in the occupancy density lead to significant changes in the indoor RF signal map, implying the need to move to a more dynamic fingerprint database.

Our results show that indoor localization techniques are not yet capable of supporting *continuous, fine-grained location tracking: the energy overheads of inertial motion estimation and continuous Wi-Fi tracking are still not low enough to permit continuous operation.* Going forward, we are working towards improving both performance and accuracy. First, we are working to exploit crowd-sourced RF measurements (from building occupants) to improve the accuracy of the Wi-Fi fingerprint database in real time. Second, we are exploring a *query-driven* location tracking paradigm, where the parameters of the location tracking algorithm are adjusted continually, on an individual-basis, taking into account the fidelity needs of the applications that use such location context.

References

[1] "Livelabs Urban Lifestyle Innovation Platform," http://www.livelabs.smu.edu.sg.

[2] M. Youssef and A. Agrawala, "The Horus Location Determination System," *Wireless Networks*, pp. 357–374, 2008.

[3] J. Stolfi, "Personal communication," 1994.

[4] G. D. F. Jr., "The Viterbi algorithm," *Proceedings of the IEEE*, pp. 268–278, 1973.

[5] P. Bahl and V. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *INFOCOM*, 2000.

[6] B. Kusy, M. Maroti, G. Balogh, P. Volgyesi, J. Sallai, A. Nadas, A. Ledeczi, and L. Meertens, "Node density independent localization," in *IPSN/SPOTS*, 2006.

[7] H. Chang, J. Tian, T. Lai, H. Chu, and P. Huang, "Spinning beacons for precise indoor localization," in *SenSys*, 2008.

[8] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support system," in *MobiCom*, 2000.

[9] K. Muthukrishnan, S. Dulman, and K. Langendoen, "Towards a rapidly deployable positioning system for emergency responders," in *UPINLBS*, 2010.

[10] O. Woodman and R. Harl, "Pedestrian localisation for indoor environments," in *Ubicomp*, 2008.

[11] I. Constandache, X. Bo, M. Azizyan, and R. R. Choudhury, "Did you see Bob?: human localization using mobile phones," in *ACM Mobicom*, 2010.

[12] N. Kothari, B. Kannan, and M. B. Dias, "Robust indoor localization on a commercial smart-phone," in *Tech. Report CMU-RI-TR-11-27, CMU*, 2011.

[13] Y. Kim, H. Shin, and H. Cha, "Smartphone-based wi-fi pedestrian-tracking system tolerating the rss variance problem," in *PerCom, 2012*, 2012, pp. 11–19.

[14] H. Lu, W. Pan, and N. D. L. et al., "SoundSense: Scalable Sound Sensing for People-Centric Sensing Applications on Mobile Phones," in *MobiSys*, 2009.

[15] M. Azizyan, I. Constandache, and R. R. Choudhury, "SurroundSense: Mobile Phone Localization via Ambience Fingerprinting," in *MobiCom*, 2009.

[16] S. Tarzia, P. Dinda, R. Dick, and G. Memik, "Indoor localization without infrastructure using the acoustic background spectrum," in *MobiSys*, 2011.

[17] H. Liu, D. Houshang, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man and Cybernetics*, pp. 1067–1080, 2007.