On the Empirical Performance of Self-calibrating WiFi Location Systems

Daniel Turner, Stefan Savage, and Alex C. Snoeren
Department of Computer Science and Engineering
University of California, San Diego
{djturner, savage, snoeren}@cs.ucsd.edu

***Abstract*—The pervasive deployment of 802.11 in modern enterprise buildings has long made it an attractive technology for constructing indoor location services. To this end, a broad range of algorithms have been proposed to accurately estimate location from 802.11 signal strength measurements, some without requiring manual calibration for each physical location. Prior work suggests that many of these protocols can be highly effective—reporting median errors of under 2 meters in some instances. However, there are few studies validating these claims at scale, nor comparing the algorithms in a uniform, realistic environment. Our work provides precisely this kind of empirical evaluation in a realistic office building environment. Surprisingly, we find that median errors in our environment are consistently greater than 5 meters and, counter-intuitively, that simpler algorithms frequently outperform their more sophisticated counterparts. In analyzing our results, we argue that unrealistic assumptions about access point densities and underlying variability in the indoor environment may preclude highly accurate location estimates based on 802.11 signal strength.**

I. Introduction

There has long been a demand for location-sensitive applications that understand and can react to their user's physical location. For example, tasks such as asset tracking, proximity-based social networking and location-specific alerting all emerge from the ability to determine precisely where a device is located. In the outdoor environment, cell-phone handsets provide a GPS-based solution to this problem (largely motivated by E911 compliance) but such hardware is rarely found in portable computers, nor does it function well indoors. For these reasons, there have been many attempts to exploit 802.11 WiFi radio infrastructure (nearly ubiquitous in indoor enterprise environments) for the same purpose.

Indeed, over the last decade many dozens of indoor client location systems have been proposed that use only the integral 802.11 radios in mobile devices and infrastructure access points. Some approaches use extensive manual calibration to build up databases of 802.11 signal fingerprints paired with geographical information, but these are highly expensive to deploy (and inherently fragile to environment change) precisely because of their labor intensive nature [9]. Thus, most modern systems attempt to be self-calibrating—only using measurements from access points in known locations and generalizing them to fit a model of RF propagation or proximity. However, most of these algorithms have been tested only in small limited environments and generally not in comparison to one another. In fact, the access point (AP) density of the published experimental environments varies over two orders of magnitude, frustrating any attempt at principled comparison. Hence, it remains unclear how effective these systems are in a practical setting. This paper is focused on precisely this question.

We evaluate the effectiveness of a representative set of self-calibrating algorithms in a large, real-world deployment. We conduct measurements within a four-story office building wired with professionally installed 802.11 access points sited for production use. Our test environment is also equipped with passive wireless monitors that allow for monitoring of all 802.11 packets from multiple vantage points [4]. We divide the literature on self-calibrating location algorithms into three categories and test a representative algorithm from each category. Beyond quantifying the absolute and relative performance of these location systems, our test infrastructure allows us to observe and report upon the shortcomings of various propagation models, limited and inaccurate training data, and several fundamental issues that impact the accuracy of all wireless positioning systems.

Our work makes three key contributions: first, we empirically evaluate the accuracy of a representative set of self-calibrating location algorithms in a realistic, independent environment: a real-world, building-wide 802.11 network. Second, we identify systematic failures in these approaches, and discuss which can likely be solved and which cannot. Finally, we identify the strengths and weakness unique to each of the systems we test and what reasonable accuracy expectations might be in production settings.

This paper is organized as follows. Section II places our work in the broader scope of 802.11-based location systems. Section III presents our taxonomy of self-calibrating algorithms and the specific algorithms we consider from each category. Section IV describes our testing methodology as well as results. Section VI explains the underlying reasons behind the unexpected results from IV. Finally, VII summarizes our findings and discussion.

II. Background

While WiFi localization systems are among the most popular, systems have been developed to exploit a wide range of technologies including Bluetooth signals [12], RFID proximity [19], and ultrasonic acoustic waves [3], among others.

Systems that use signal strength measurements, or, more specifically in the case of WiFi, the received signal strength indication (RSSI)—a unit-less measure of signal power associated with a packet as measured by a receiving device—can be subdivided into two categories: those that require manual calibration and those that do not. While the goal of this paper is to evaluate self-calibrating 802.11-based location systems, we briefly summarize the approaches based upon manual calibration for context.

Many of the early (and still most accurate) 802.11-based location systems require manual calibration. During this initial site survey an installer collects client fingerprints (RSSI values for packets broadcast by APs) at a large number of known locations to build up a fine-grained fingerprint database that can be used for future client localization. In general these systems have three major manifestations: point matching, probabilistic matching, and Bayesian networks.

Point matching, also known as *nearest neighbor in signal strength* [2], starts by capturing a client fingerprint and comparing it against the initial fingerprint database scanning for the fingerprint that minimizes a similarity metric, often Euclidean distance. The client's location is determined to be the location of the offline fingerprint that minimizes the metric.

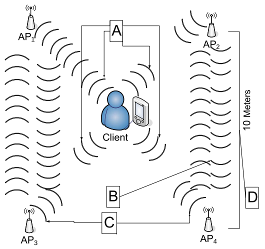
The second set of methods once again starts with the user taking a client fingerprint. Then the user computes the probability of receiving the fingerprint at each location, $l = (x, y)$, in the building. Assuming that the distribution of RSSIs in the fingerprint is Gaussian, it is straightforward to determine an l such that $P(s|l)$ is maximized [1], [25].

Algorithms based on Bayesian networks are the most complicated. They depend on building a Hidden Markov Model (HMM) in which the RSSIs represent the observed state, and the distance from the APs (and hence the x, y coordinates of the client) are the hidden state. The relationship between visible and hidden states is encoded via a signal decay model and then the fingerprint database is used to train the parameters of the model [16], [21].

Chintalapudi et al. recently proposed *EZ Localization* as a methodology for calibration-free wireless localization in an indoor environment [6]. *EZ* uses WiFi-enabled smart phones to take fingerprints at unknown locations in a building. Rather than using the location of APs to ground their model, however, they leverage the GPS units available on the client devices. When a phone is able to get a GPS lock they opportunistically use the measurement as an anchor point. They then treat the fingerprints as constraints to be solved by a genetic algorithm. The authors report that *EZ* has low error in their environment, but remains unable to surpass the accuracy of systems that require manual calibration.

III. Self-Calibrating Algorithms

Because manually calibrated systems may require taking hundreds of fingerprints per floor, a second class of systems that requires no manual calibration has been proposed. Instead, these *self-calibrating* systems leverage information regarding the location of WiFi access points to conduct a training phase in which they collect system fingerprints, or sets of RSSI measurements taken of (and by) the stationary APs in the network. After the self-calibration stage, the fingerprint database is combined with pre-configured information about the physical location of the APs to compute a model of the relationship between RSSI and physical location. This model is then used to answer on-line queries about client location.



(a) Simplified network diagram

Device	Mean RSSI
AP_1	24.4
AP_2	47.8
AP_3	45.5
AP_4	40.6

(b) Client fingerprint

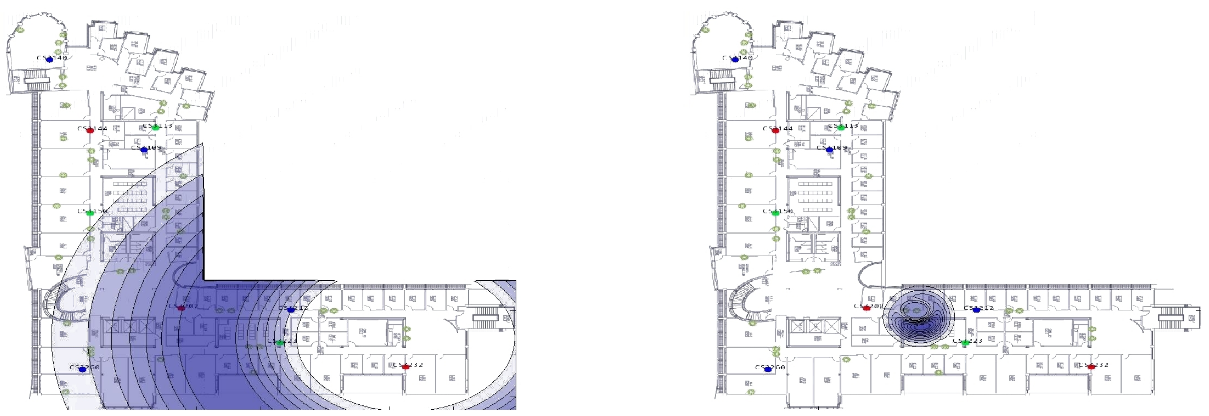
Fig. 1. An example network and fingerprint corresponding to signal A

A. Overview

It is convenient to think of self-calibrating algorithms as fitting into three categories based on the amount of data used to perform calibration. To assist the reader in understanding each algorithm we give examples of how they work in a simple wireless network. Our explanatory network, diagrammed in Figure 1(a), has four APs and one client. Each AP knows the distance between it and every other AP, for example the distance between AP_3 and AP_4 is ten meters and labeled D. The 802.11 signal emitted by the client when sending a packet is labeled A. The RSSI values for one transmission as seen by the four APs, i.e. a client fingerprint, are documented in the table in Figure 1(b). Conversely, the 802.11 signal generated when an AP sends a packet into the network is labeled C. When an AP sends a packet it is heard by everyone within range, though all but the intended destination will usually ignore it when the network is operational. In the system training phase, however, APs do not ignore packets sent from other APs, labeled B, but instead record their RSSI values. Table I shows an example of training data that might observed by AP_4 in this network.

Baseline: In order to provide some context in which to interpret the performance of the localization system classes, we consider a simple, calibration-free baseline that leverages only a single RSSI from a client transmission, labeled A in Figure 1(a). This nearest-neighbor approach requires no calibration: to locate a client it simply determines which AP heard the client's transmission with the strongest RSSI and declares the client to be co-located with that AP. For example if the system obtained the fingerprint in Figure 1(b) it would estimate the client to be at the same location as AP_2.

Class I: Our baseline algorithm, while simple, has two major limitations. First, it can choose only locations occupied by an AP. Second, it ignores all information from the $(n-1)$



(a) AP_4's RPM

(b) Final RPM derived from individual AP RPMs

Fig. 2. Example RPMs. Darker shading indicates higher probability.

APs that do not observe the strongest RSSI. To address the first issue *probabilistic methods* use data from the fingerprint database, gathered during the training phase, to train a model of decay that allows them to map an RSSI to locations other than the location of an AP. The probabilistic methods also use all n observed RSSIs when computing a final location.

Here, we consider the probabilistic algorithm proposed by de Moraes and Nunes [8] due to its reported mean error of less than two meters and its foundation on a well-cited and understood RF propagation model [20]. The first step in the client location phase of this algorithm is to estimate the distance between each AP and the client. To do this each AP that hears the client builds a *radio propagation map* (RPM). An RPM is best thought of as a grid overlaid on the building's floor plan where each grid point, $l = (x, y)$, has an associated probability, $P[l|s_i]$, that the client is that location when signal strength s_i is observed by AP i. $P[l|s_i]$ is assumed to be a Gaussian probability distribution, based on a large scale decay model where the model is trained on one entry in the offline training database, Table I. An overly simple way to view the RPM is that the more similar an observed RSSI at location l is to the value the large scale decay model predicts, the more likely $P[l|s_i]$ will be large.

Once the probabilistic method has an RPM for each of the n APs it combines them in the obvious manner, $P[l|s] = \Pi_{i=1}^{n} P[l|s_i]$ Finally a quick scan of the joint RPM is performed to find the point with the highest probability and that point is declared as the client's location. For example, if the algorithm's training phase produced Table I and subsequently observed the client fingerprint in Figure 1(b), Figure 2(a) shows the RPM AP_4 generates overlaid on our building's floor plan while Figure 2(b) depicts the final RPM produced when all of the four RPMs are combined into a final RPM.

Class II: The probabilistic algorithms leverages more information than our nearest-neighbor baseline: During the training phase, each AP observes $(n-1)$ different RSSI-distance pairs, i.e., AP_4 observes the RSSI values from AP_1, AP_2, and AP_3. But, when an AP creates an RPM it uses only one RSSI/distance. Algorithms in our second class overcome this limitation of probabilistic algorithms by using the reported signal strength at multiple APs simultaneously when attempting to locate the client [10], [14], [18]. We have chosen Triangular Interpolation and eXtrapolation (TiX) [10] as the representative example for our second class of algorithms.

Device	Distance (m) from sender	Mean RSSI	Variance
AP_1	4.6	48.9	1.2
AP_2	10.0	42.3	0.8
AP_3	15.4	22.6	0.2

TABLE I
AN EXAMPLE OF AP_4'S OFFLINE TRAINING DATABASE.

As before, the first step in TiX's training phase is for each AP to record the average RSSI of the other APs, i.e. build the fingerprint database shown in Table I. Each row consists of the distance to the indicated AP as well as the mean and variance of the signals between them (labeled D and B, respectively, in Figure 1(a)). In the second phase each AP fits these values to an exponential decay function of the form $f(x) = a \times e^{bx}$, where x is the RSSI, $f(x)$ is the computed distance. This function gives each AP a mapping between RSSI and distance that uses all of the information available to it during the training phase.

Once the algorithm is trained it can begin to locate devices. To locate a device it needs a packet to be heard by three APs. (We discuss the case where a client is heard by more than three APs in Section IV-C2.) It then applies the RSSI to distance function at each of the APs. Finally, it performs a modified version of trilateration. The modified version of trilateration handles the case where noisy measurements cause there to not be a single intersection of the radii of all the three circles.

For example, if TiX were to locate the client that produced the fingerprint in Figure 1(b) it might use APs 2, 3, and 4. Further, if TiX used the training data in Table I it would find that the RSSI to distance curve for AP_4 is $f(x) = 69.2e^{-0.056x}$ and it would place the device 6.9 meters away from AP_4. AP_2 and AP_3 compute similarly.

Class III: The final class of algorithms we study makes one significant conceptual addition to the third class of algorithms: It recognizes that each AP is able to map the distance from itself to the client via RSSI, but, also, that the other $(n-1)$ APs can map the distance between the client and AP from their own perspective [11], [17]. For example, AP_4 can not only estimate the distance between itself and the client, but also the distance between the client and the other APs. Thus final estimation of the distance between AP_4 and the client is a (linear) combination of each AP's distance estimation between the client and AP_4. We chose ZCFG [17] as our representative algorithm for Class III because of its high level of sophistication and built-in resistance to error from environmental noise.

ZCFG, like the previous algorithms, requires a training phase during which each AP records the average RSSI for all APs in range. Unlike the TiX algorithm, the calibration of ZCFG is done in real time. ZCFG uses a *signal distance map* (SDM) to "describe the relation between the RSS(I)s and the geographic distance" [17]. Every time ZCFG wants to locate a device, it identifies each of the n APs that observed a packet from the device. It then builds the $n \times n$ dimensional SDM according to the following equation, $T = \log(D)S^T(SS^T)^{-1}$ where S, D are $n \times n$ dimensional matrices such that S_{ij}, D_{ij} are the RSSI as observed by AP i from AP j and the physical distance between AP i and j respectively. ZCFG then finds the distance between the device and each of the n APs according to the equation: $d_n = e^{Ts_n}$ where s_n is the column vector of the observed RSSIs. Finally, ZCFG uses a multilateration algorithm to determine a location estimation.

IV. Experimental Evaluation

Our goal is to perform a fair and accurate head-to-head comparison of the three exemplar algorithms presented in the previous section. We start by quantifying the performance of the algorithms in several locations in our office building. More importantly, we attempt to understand limiting factors that will effect most real-world deployments. We also identify several practical issues that were not discussed by the algorithms' authors in their original publications and, where possible, offer guidance in addressing them.

A. Experimental setup

Our test building has approximately 145,000 square feet over four stories and a basement. All experiments were conducted on the third floor which has ten access points divided between channels 1, 6, and 11.

The APs in our building are not under our control. This means that we are unable to use them to observe traffic. Instead, we monitor traffic using passive wireless monitors [4]. There are 48 pairs of such monitors installed throughout our building, with approximately 12 pairs (24 distinct monitoring devices with two radios each) located on each floor. Like the initial AP placement, the monitor layout was specifically designed to provide a maximal coverage of each floor and, like the APs, should neither favor nor disfavor any specific location. The locations of the APs and monitors are indicated in Figure 3.

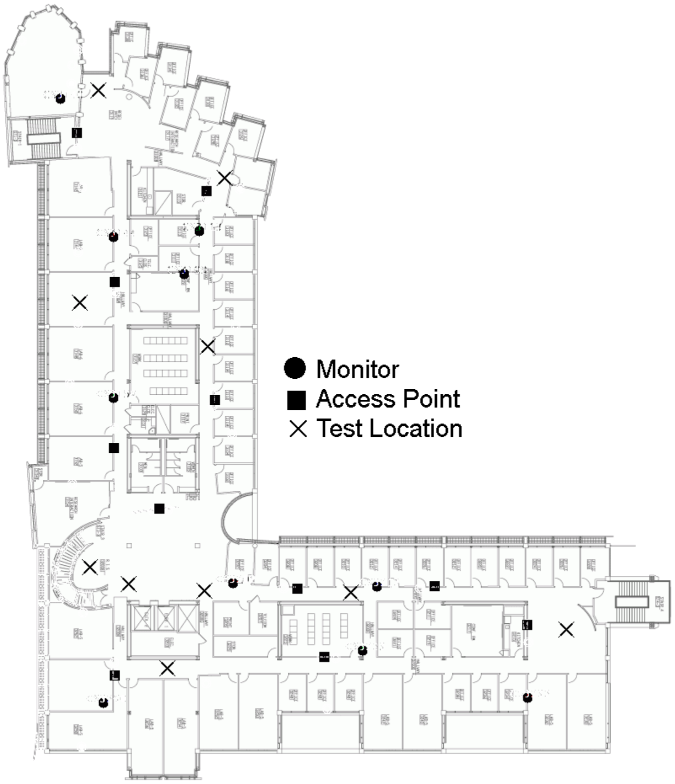


Fig. 3. Locations on the 3rd floor

For *training* the algorithms, we opportunistically use the existing AP *traffic* (i.e., as though each AP was a client) as we know the precise location of each AP. This is sufficient for constructing the fingerprint database for the trivial, probabilistic, and TiX algorithms. Since the online training process of ZCFG requires an architecture in which packets are transmitted as well as received, we implement this by forcing each monitor to send pings to each of the others. Since there are 11 pairs of monitors but only 10 APs on the floor we consider, ZCFG will, in principle, have more training data to work with. This should give ZCFG an advantage over the other three algorithms, but Section IV-D shows that this advantage does not materialize.

To build the fingerprint database we allow each algorithm to monitor the broadcast packets sent from the APs in the building for approximately ten seconds. We find that there is sufficient traffic in ten seconds to allow the mean RSSI value to be minimally affected by abnormally large or small instantaneous values. It is possible to locate a client from only a single packet. However, we averaged two seconds worth of RSSI values from the client. This time frame is a balance between reducing the effect of an outlier in the RSSI values and effectively locating a user even if they are in motion. Throughout this paper we report RSSI values instead of dB; RSSI values can be converted to dB by subtracting 95.

Experiment	Square Feet	# of APs	APs/Sq. Ft.
Ours	30,000	10	0.0003
de Moraes & Nunes (Prob.)	1,722	4	0.0023
Gwon & Jain (TiX)	6,717	4	0.0006
Lim *et al.* (ZCFG)	598	6	0.01

TABLE II
COMPARING PHYSICAL TESTBED ENVIRONMENTS

Method	**Accuracy (m)**			
	Min.	**Median**	**95%**	**Published median**
Nearest (I)	0.9	6.2	12.3	n/a
Probabilistic (II)	0.7	5.8	20.7	1.84 [8]
TiX (III)	2.9	7.8	14.8	5.0 [10]
ZCFG (IV)	2.7	12.4	37.5	2.5 [17]

TABLE III
MINIMUM, MEDIAN, 95%, AND PREVIOUSLY PUBLISHED ERROR IN METERS.

B. Monitor Density

One of the major contributions of our work is to empirically evaluate the accuracy of self-calibration systems in a uniform and realistic context. We evaluated these systems throughout our building including: hallways, interior spaces, as well as large and small offices on the perimeter of our building. But perhaps the biggest difference between our study and previous work is the number of APs per square foot. The ideal number of APs needed per square foot depends on many characteristics but two of the most important are building shape and wall type: brick, drywall, or open cubicle. Cisco recommends one AP per 3,000 to 5,000 square feet (i.e., a density of 0.0002–0.0003 APs/sq. ft.) [7]. Table II lists the size of the published test environment for each algorithm we study as well as the number of active APs per square foot. While the AP density in our building is on the high side of recommended (recall that the monitors are even slightly *more* dense), the testbeds used in previously presented results are between 2 and 30 times more dense than recommended. Hence, we expect the performance of these systems in our environment to degrade accordingly.

C. Practical considerations

In the course of implementing and evaluating the algorithms, we discovered that small modifications were required to two of the published approaches in order to obtain reasonable results.

1) Probabilistic model: When analyzing the data from the probabilistic model we find it helpful to add a floor to the values of $P[l|s_i]$. Allowing arbitrarily small values causes an extremely poor RPM to convince the algorithm that a given location is infeasible no matter how strongly the other RPMs believe it to be likely. We also disregard the RPM of any monitor whose observed RSSI value of the client's signal is more than one standard deviation away from any training point. The underlying reasons for and impact of these changes are discussed in Section VI-A.

2) TiX: On average a client is heard by four to six monitors. But, the TiX method uses trilateration to determine a final location and therefore requires that we choose which three monitors to use. While this issue likely did not arise frequently in the original test environment with only 4 APs, our building—and likely many enterprise environments—employs significantly more.

We consider several techniques for choosing the three monitors to use: the monitors with strongest RSSI, the monitors that provide the instantaneously best localization, the best over all time (the three monitors that perform best in general), and the three monitors physically closest to the client being located. The latter three methods require the use of an oracle which is unrealistic, but provide interesting insights regarding the best potential performance of TiX. Unsurprisingly, the instantaneous best and best over all time perform significantly better than the other methods. In particular, we find that TiX can do quite well with omniscient monitor selection—in some cases as accurate as 0.5 meters. Examining which monitors lead to the best results, we find that for 27% of all measurements the three monitors that observed the strongest RSSI are the three monitors that provide the instantaneous best result, and 40% of the time the three monitors that observed the strongest RSSI produce an error within 10% the instantaneously best localization. For practical reasons, we report only the performance for the three monitors with the strongest RSSI as it is the only implementable method. However, we find that choosing the three monitors physically closest to the laptop is not any better than simply choosing the three monitors that observed the strongest RSSI. This observation is further reinforced by the fact that for 40% of all measurements at least one of the three monitors providing the instantaneously best localization observes an RSSI under 25, implying that one monitor is more than 15 meters away from the laptop.

D. Results

We measure the accuracy of each system in our test environment, where accuracy is defined as the distance between the estimated location and actual location. For each run of an experiment all of the algorithms attempt to locate a stationary Apple MacBook Pro every fifteen seconds over a continuous three-minute period. The computer was on a stand approximately 1.5 meters tall. While we did not control for the orientation of the laptop an effort was made to keep people away from the laptop during each run. Location detection was attempted in fifteen different locations. A subset of the test locations on the third floor can be seen in Figure 3. We expect the accuracy of the naive, nearest-AP baseline to be the poorest, and that accuracy will increase as we move from Class I to III.

Table III lists the minimum, median, and 95th-percentile error in our building for the nearest monitor, probabilistic method, TiX, and ZCFG. Table III also lists the previously reported median error for each of the probabilistic, TiX, and ZCFG methods as reported from the literature in which they were introduced. Immediately, we see that the measured error in our environment is much higher than the published errors. We also notice that the that two simplest algorithms both do better on average than TiX and ZCFG with the probabilistic algorithm doing the best overall.

Method	Floor	1.5m	Ceiling
Nearest	6.7	6.7	4.8
Probabilistic	6.6	3.37	2.65
TiX	16.8	11.6	11.2
ZCFG	9.5	14.2	12.9

TABLE IV
MEDIAN ERROR IN METERS AT DIFFERENT HEIGHTS FOR A SUBSET OF THE LOCATIONS.

Also surprising is the large 95% error. For the nearest-AP method, the error is due to the fact that one test location is approximately 12 meters from the nearest monitor. The remaining methods are not limited by the distance between a monitor and the laptop, however. For these methods, localizations with the highest error rate occur when one or more of the monitors does not hear the laptop, even though the laptop was heard in previous rounds. This requires the localization methods to use the remaining—though less ideal—monitors. This phenomena has been previously studied and is likely due to obstacles such as people traversing our building [13].

V. CAUSES OF ERROR

Given that all of the algorithms perform significantly worse than their previously reported accuracy, we investigate how much of this performance mismatch is due to environmental factors. Previous studies of outdoor and large-scale environments motivate us to focus our search on two issues: signal reflection from walls and RSSI variations over periods of time [5], [22], [24].

A. Signal Reflection

One known reason for unexpected RSSI decay is reflection from floors [21]. Luckily, the APs and monitors in our building are mounted at ceiling height and therefore are unlikely to be significantly affected by ground reflection. But if floors can cause RSSI decay, it seems plausible that the ceiling can as well. This is especially worrisome given that our calibration data is measured at ceiling height while most people—and our experiments—operate wireless devices at chest level. We therefore run controlled experiments in three of the previous test locations, as well as directly under Monitor 3115, with the laptop on the ground, on a 1.5-m stand, and at ceiling height.

The median errors at each of these four locations are presented in Table IV. The error when the laptop is on the floor is much greater than at the ceiling or at a height of 1.5 m. But there is also a difference in accuracy between the stand and the ceiling. To help illuminate this difference, Figure 4 plots RSSI vs. time for each of the monitors, depicting the three common classes of result. In Figure 4(a) all three heights show very little difference. However, the other two figures show that for some locations monitors hear the laptop much more clearly at ceiling height while others hear best at the stand height. In Figure 4(c) the difference between the ceiling and the stand is on average 4 RSSI or up to 4 meters according to Monitor 3206's model. While we do not know exactly what causes the three scenarios in Figure 4, it is likely due to multipath effects. The median errors across four location, shown in Table IV, indicate that height disparity can be responsible for approximately one half a meter to two meters of error.

B. Time Variation

Studies have shown that RSSI readings of a stationary device can have a large variability over relatively short time scales [15], [23]. Conveniently self-calibrating systems are implicitly easy to recalibrate. It is not clear how often one actually needs to recalibrate, however. To determine how damaging stale calibration data is we captured AP broadcast traffic: every ten minutes for an hour, once an hour for a day, once a day for two months, and once a month for seven months. The hourly, weekly, and monthly data collection was collected at 11am. For the monthly data collection we always collected data on a weekday near the beginning of the month.

Figure 5 shows the worst case results of a representative monitor running TiX's calibration across each of the time periods above. Specifically, for each RSSI value we show the the minimum and maximum distance for which it could be mapped. For convenience we also show the result of training on all samples.

As we can see the hour-long time window shows that there is virtually no deviation within an hour. On the other hand, in the seven-month study the bulk of the RSSI values can be mapped to distances as far as 5 meters apart by the two most extreme periods. Thus, infrequent recalibration will have a disproportionate impact on localization at longer distances. The tails of all four graphs in Figure 5, however, show that low RSSI values are universally poor; e.g., an RSSI of 15 can be mapped anywhere from 27–35 m regardless of time scale.

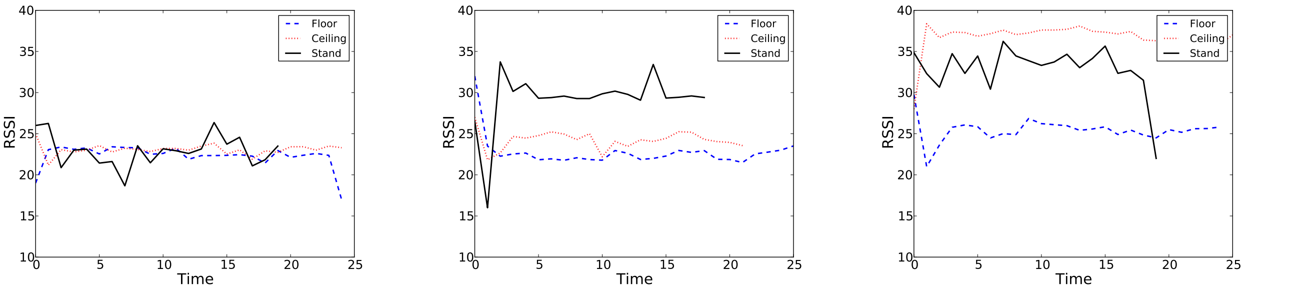
VI. DISCUSSION OF SYSTEMATIC FAILURES

The accuracy observed in our building for each algorithm is much worse than previously reported. While some of this error can be attributed to environmental factors, some cannot. In this section we offer some observations that apply to all self-calibrating systems in our environment

A. Extrapolation Failure

One of the most obvious differences between self-calibrating systems and those that require offline manual calibration is the number of entries in their respective fingerprint databases: The fingerprint databases for the self-calibrating systems in our experiments contain entries for less than a dozen locations as compared to the hundreds of entries required in manually calibrated systems. In our experience, it is frequently the case that this limited amount of information, i.e. the number of observed RSSIs in each fingerprint (i.e., rows in Table I), is not sufficient to accurately represent the actual RSSI decay in the building. In particular, the resulting model is often unable to predict the correct decay at distances further then its furthest training AP or closer than its closest training AP. We have termed these issues *extrapolation failures*.

Section IV-C1 notes that we modify the probabilistic algorithm to reject RPMs that are based off of a training point more than one standard deviation from the observed client RSSI. This filtering step reduces the mean error from 11.3 to 5.0 m, and the median from 7.3 to 5.2 m as reported in Table III. Before we made this change it was very common

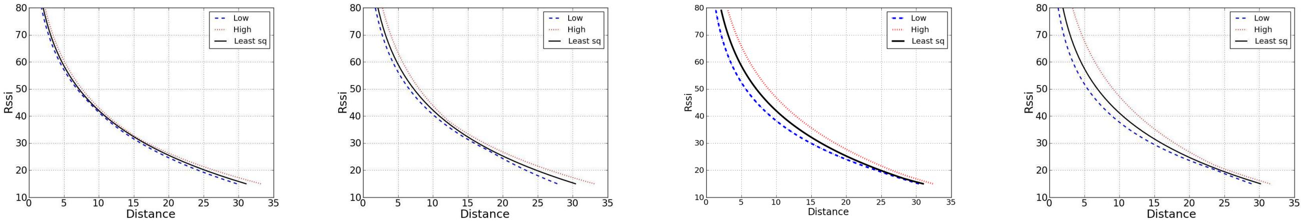


(a) Location 4: Monitor 3108 observes similar RSSIs regardless of client height

(b) Location 2: Monitor 3115 observes the strongest RSSI when the client is near stand height

(c) Location 4: Monitor 3206 observes the strongest RSSI with client near the ceiling

Fig. 4. Observed RSSI values at three different heights in three distinct test locations.



(a) Sampling once every ten minutes

(b) Sampling once every hour

(c) Sampling once every day

(d) Sampling once every month

Fig. 5. RSSI variability as shown by fitting exponential decay curves to monitor 3108's most extreme datum over different intervals

to see RPMs that effectively believed the device could be no further than one meter away from the monitor. In these situations it was also common to see that the closest AP a monitor trained on would be only a few meters away but, also be on the other side of a wall. This wall would cause the RSSI from the AP to be degraded. However, when the client is at approximately the same distance as the closest training AP but not separated by a wall, the observed RSSI would be much stronger, often more than 10 units. This additional 10 units of RSSI over the nearest AP combined with the model's assumption of exponential decay would place the client far too close to the monitor.

The opposite problem is also common. The furthest AP a monitor would train on would be separated by several walls, and, thus, have a degraded RSSI. But a client located further away than the furthest AP the monitor trained on might have line-of-sight to the monitor due to a long hallway. Because the client has line-of-sight its RSSI is much stronger than the furthest training AP. This causes RPMs where the monitor believes the client is much closer than it actually is. One might hope to overcome this problem by learning to pick better RSSI-distance pairs from fingerprints. Unfortunately, it is not clear how to choose a "best" RSSI-distance pair.

The third class of algorithms, in contrast, uses *all* of the RSSI-distance pairs in a fingerprint when calibrating. Unfortunately, this does not eliminate all extrapolation failures in TiX. For example, when locating position 5, in the lower left hand corner of Figure 3, the three monitors that observe the strongest RSSI are 3213, 3216, and 3223. The first two monitors have only minimal error when mapping RSSI to distance at this location, but the same is not true of the third. If we look at the third monitor's training data we observe that the furthest training point is 25.3 meters away and has an RSSI of 20. Monitor 3223's mapping curve places our client's RSSI of 26.2 at a distance of 17.5 meters when the client is actually 28.8 meters away. We observe similar instances when the RSSI value is stronger than any of our training points.

The examples above show that extrapolation failures are fundamentally due to the limited amount of information in each fingerprint. These failures are not inherent to the models, but instead to the limited amount of training data self-calibrating systems collect.

B. Interpolation Failure

Extrapolation failures could likely be overcome by adding a few additional APs at the extremes of the building. We simulate such an environment by manually removing the RPMs that suffer from extrapolation failure from our test cases. We term the remaining erroneous cases *interpolation failures*: An interpolation failure occurs when a model is unable to accurately map RSSI to distance even though the correct distance is between the closet and furthest AP.

We find that interpolation failures typically arise when a client is in a qualitatively different position than the best-fit training point, e.g. the client's position is closer to the monitor but is separated by more walls. The differing number of walls cause the RSSI to decay more slowly or quickly than the monitor expects and, as a result, the monitor inaccurately locates the client.

One might hope that more sophisticated algorithms are better able to avoid interpolation failures. Surprisingly, we

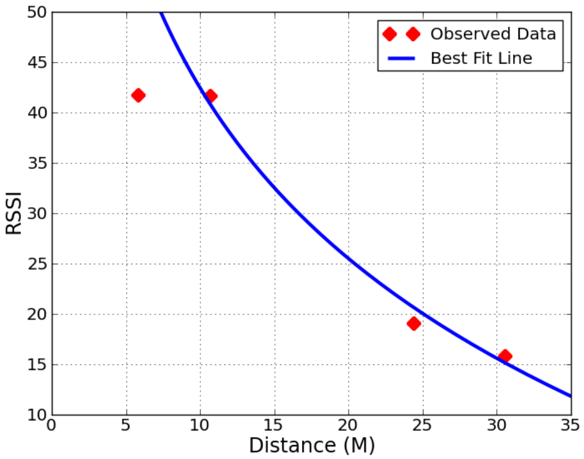


Fig. 6. Monitor 3206a exhibits interpolation failure.

find the inverse is true. In particular, TiX is more sensitive to training data that is not representative of client locations than the probabilistic method. For example, Figure 6 plots both the training data and the best-fit curve for Monitor 3206a. We see that at distances of 5.8 m and 10.7 m the observed RSSI values are 41.7 and 41.6. The least-squares fitting strategy then causes RSSIs of 41.7 and 41.6 to map to distances of 10.3 m and 10.2 m, respectively. In contrast, RSSIs of 55.7 and 40.7 are required for the best-fit curve to believe we are at a distance of 5.8 m and 10.7 m respectively. The end result is an insufficiently concave curve that requires large changes in RSSI to make small changes in its distance estimation. TiX also suffers from the reverse instantiation of the failure condition above. In this case the underlying data causes the decay to be too convex. This happens when there is a large difference between the RSSI values of two data points but not their distance. In either case adding additional APs will not solve the problem.

The underlying issue is that there is a many-to-many relationship between RSSIs and distance from a monitor and the monitor lacks contextual clues to allow it to determine the best mapping. Thus, *interpolation failures* cause noticeable decreases in accuracy and are also a fundamental issue that can not be solved by simply increasing model complexity and are only partially mitigated by increasing the amount of information per fingerprint in the fingerprint database.

C. More Noise than Signal

As each class of algorithm we discuss uses more data to train we would expect increased accuracy. However, Table III clearly contradicts our assumption since ZCFG is the least accurate. We have seen that both the probabilistic and the TiX algorithms suffer from both extrapolation and interpolation failures. Ideally we would like to show that ZCFG also suffers from the same issues. Unfortunately, ZCFG uses so much training data at each step that we are unable to attribute specific instances of failure to only one or two RSSI values in the fingerprint database. Instead, we show two major limitations to improving the accuracy of ZCFG.

First, recall that both the probabilistic method and TiX makes a dynamic choice on how to calibrate their models based on the observed RSSI of a client. Unfortunately, in ZCFG the SDM is computed independently of the observed RSSI of the client. As a result the scale factors of each entry of the SDM are computed ahead of time and the same linear combinations of weights are used to locate a client in any part of the building. Thus, ZCFG's requirement to optimize for locating a client in any part of the building removes its potential ability to detect and ignore parts of its model that could be inaccurate in a given situation. As a result ZCFG is highly vulnerable to extrapolation and interpolation failures.

In the probabilistic algorithm we discovered that if an observed RSSI was too dissimilar to anything observed in the training data its resulting RPM is very likely to suffer from either interpolation or extrapolation issues. We therefore explored if a filtering technique applied to ZCFG could tell us which observations not to use. We did this in two ways, first was that any monitor which observed an RSSI too dissimilar to anything in its training data was simply ignored. The second technique was to allow the monitor to be used in creating the matrix T but was ignored during multilateration. Totally ignoring a monitor is a actually a poor choice and pushes the average error to 18 meters. This large decrease in accuracy stems from the fact that totally ignoring a monitor removes a significant amount of useful training information. However, it is surprising that removing the monitors from multilateration fails to improve our results and actually decreases accuracy by half a meter. Further investigation shows that this filtering technique does remove some poor distance estimations. But in general it let many poor distance estimations remain as well as removing some accurate distance estimations. Therefore our simple filtering mechanism is no longer a good predictor.

In summary, the potential advantage of ZCFG is that each distance estimation is actually a linear combination of n distance estimations. Unfortunately, ZCFG is unable to identify when some of the distance estimations suffer from either an extrapolation or interpolation failure, and, worse the obvious techniques to filter out instances where either failure occurs are not effective. As a result, even the most sophisticated model we study is unable to overcome the lack of fidelity in the fingerprint database in our environment.

VII. Conclusions

We have evaluated the performance of self-calibrating 802.11-based location algorithms in a realistic office building environment. Like many previous research efforts our experiments have been conducted in one physical environment, but our building is far more representative of typical commercial buildings than the initial test environments for these systems. We tested several algorithms representative of different popular approaches: a probabilistic method, Triangular Interpolation and eXtrapolation, and ZCFG, as well as comparing against a simple nearest neighbor method. Empirical results were uniformly worse—at times much worse—than the reported performance of each algorithm.

We find that systemic problems can result from environmental issues. For example, the placement of APs at ceiling height while client usage occurs much closer to the floor creates a height imbalance that, in turn, causes non-linear RSSI degradation that cannot be corrected online. While some of the error is clearly due to systemic issues, the different results between algorithms implies that there are also failures specific to the algorithms themselves. The systems based on modeling decay are fragile due to limited fidelity in the fingerprint database combined with the inability of models to compensate for the walls and other obstacles between client and fixed radios. Surprisingly, simpler models are more robust because they are somewhat able to determine the "confidence" of a mapping in light of available data, and therefore allow for easy filtering of low confidence estimations.

Therefore, our experience is that simple models that provide a method to determine confidence in a distance estimation are more accurate than the more complex models that cannot differentiate between good and bad distance estimations. In the end even the most sophisticated models of decay are unable to overcome the challenges above. We therefore conclude that achieving accuracy with less than five meter of error in office buildings that have 802.11 APs deployed at manufacturer-recommended densities may be unrealistic with self-calibrating systems.

References

[1] A. Agiwal, P. Khandpur, and H. Saran, "LOCATOR: location estimation system for wireless LANs," in *WMASH*, 2004.

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

[3] G. Borriello, A. Liu, T. Offer, C. Palistrant, and R. Sharp, "Walrus: wireless acoustic location with room-level resolution using ultrasound," in *MobiSys*, 2005.

[4] Y.-C. Cheng, J. Bellardo, P. Benko, A. C. Snoeren, G. M. Voelker, and S. Savage, "Jigsaw: Solving the puzzle of enterprise 802.11 analysis," in *SIGCOMM*, 2006.

[5] Y.-C. Cheng, Y. Chawathe, A. Lamarca, and J. Krumm, "Accuracy characterization for metropolitan-scale Wi-Fi localization," in *Mobisys*, 2005.

[6] K. Chintalapudi, A. Padmanabha Iyer, and V. N. Padmanabhan, "Indoor localization without the pain," in *MobiCom*, 2010.

[7] *Deployment Guide: Cisco Aironet 1000 Series Lightweight Access Points*, 78th ed., Cisco Systems, Inc., 2005.

[8] L. F. M. de Moraes and B. A. A. Nunes, "Calibration-free WLAN location system based on dynamic mapping of signal strength," in *MobiWac*, 2006.

[9] E. Elnahrawy, X. Li, and R. Martin, "The limits of localization using signal strength: a comparative study," *IEEE SECON*, Oct. 2004.

[10] Y. Gwon and R. Jain, "Error characteristics and calibration-free techniques for wireless LAN-based location estimation," in *MobiWac*, 2004.

[11] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal, "Ariadne: a dynamic indoor signal map construction and localization system," in *Mobisys*, 2006.

[12] A. Kotanen, M. Hannikainen, H. Leppakoski, and T. D. Hamalainen, "Experiments on local positioning with bluetooth," *ITCC*, 2003.

[13] D. Kotz, C. Newport, and C. Elliott, "The mistaken axioms of wirelessnetwork research," Dept. of Computer Science, Dartmouth College, Tech. Rep., July 2003.

[14] A. S. Krishnakumar, A. S. Krishnakumar, W. hua Ju, C. Mallows, and S. Ganu, "A system for lease: Location estimation assisted by stationery emitters for indoor rf wireless networks," in *IEEE Infocom*, 2004.

[15] J. Krumm and E. Horvitz, "LOCADIO: inferring motion and location from Wi-Fi signal strengths," *MOBIQUITOUS*, 22-26 Aug. 2004.

[16] A. M. Ladd, K. E. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. S. Wallach, "Robotics-based location sensing using wireless ethernet," in *MobiCom*, 2002.

[17] H. Lim, L.-C. Kung, J. C. Hou, and H. Luo, "Zero-configuration, robust indoor localization: Theory and experimentation," *INFOCOM*, 2006.

[18] D. Madigan, E. Einahrawy, R. Martin, W.-H. Ju, P. Krishnan, and A. Krishnakumar, "Bayesian indoor positioning systems," in *INFOCOM*, 2005.

[19] S. Polito, D. Biondo, A. Iera, M. Mattei, and A. Molinaro, "Performance evaluation of active RFID location systems based on RF power measures," *PIMRC*, 2007.

[20] T. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[21] T. Roos, P. Myllymki, H. Tirri, P. Misikangas, and J. Sievnen, "A probabilistic approach to WLAN user location estimation," *International Journal of Wireless Information Networks*, vol. 9, 2002.

[22] T. Stoyanova, F. Kerasiotis, A. Prayati, and G. Papadopoulos, "Evaluation of impact factors on RSS accuracy for localization and tracking applications," in *MobiWac*, 2007.

[23] P. Tao, A. Rudys, A. M. Ladd, and D. S. Wallach, "Wireless LAN location-sensing for security applications," in *WiSe*, 2003.

[24] K. Whitehouse, C. Karlof, and D. Culler, "A practical evaluation of radio signal strength for ranging-based localization," *ACM MC2R*, 2006.

[25] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *MobiSys*, 2005.