Indoor Localization Without the Pain

Krishna Chintalapudi
Microsoft Research India
krchinta@microsoft.com

Anand Padmanabha Iyer
Microsoft Research India
v-anandi@microsoft.com

Venkata N. Padmanabhan
Microsoft Research India
padmanab@microsoft.com

ABSTRACT

While WiFi-based indoor localization is attractive, the need for a significant degree of pre-deployment effort is a key challenge. In this paper, we ask the question: *can we perform indoor localization with no pre-deployment effort*? Our setting is an indoor space, such as an office building or a mall, with WiFi coverage but where we do *not* assume knowledge of the physical layout, including the placement of the APs. Users carrying WiFi-enabled devices such as smartphones traverse this space in normal course. The mobile devices record Received Signal Strength (RSS) measurements corresponding to APs in their view at various (unknown) locations and report these to a localization server. Occasionally, a mobile device will also obtain and report a location fix, say by obtaining a GPS lock at the entrance or near a window. The centerpiece of our work is the *EZ Localization* algorithm, which runs on the localization server. The key intuition is that all of the observations reported to the server, even the many from unknown locations, are constrained by the physics of wireless propagation. EZ models these constraints and then uses a genetic algorithm to solve them. The results from our deployment in two different buildings are promising. Despite the absence of any explicit pre-deployment calibration, EZ yields a median localization error of 2m and 7m, respectively, in a small building and a large building, which is only somewhat worse than the 0.7m and 4m yielded by the best-performing but calibration-intensive Horus scheme [29] from prior work.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**]: Miscellaneous

General Terms

Algorithms, Design, Experimentation, Theory

1. INTRODUCTION

The need for location information to enable pervasive computing applications in indoor environments, coupled with the unavailability of GPS in such environments, has motivated a large body of research on indoor localization. In particular, there has been a focus on leveraging existing infrastructure (*e.g.*,WiFi access points) to enable indoor localization, the advantage being that the cost of deploying a specialized infrastructure for localization is avoided. Existing solutions, however, require extensive pre-deployment effort, for instance, to build detailed RF maps [4] or RF propagation models based on surveys of the environment. *In this work, we propose a novel indoor localization system, EZ, that leverages existing infrastructure without requiring any explicit pre-deployment effort.*

EZ relies on three basic assumptions: (i) that there are enough WiFi APs to provide excellent coverage throughout the indoor environment, (ii) that users carry mobile devices, such as smartphones and netbooks, equipped with WiFi, and (iii) that occasionally a mobile device obtains an absolute location fix, say by obtaining a GPS lock at the edges of the indoor environment, such as at the entrance or near a window. In EZ, users simply sit, stand, or move around in the indoor environment in normal course. While they do so, each user's mobile device records the received signal strength (RSS) from the WiFi APs visible to it at various (unknown) locations, and reports this information, along with the occasional location fix when available, to a central localization server. The server uses this data to simultaneously learn the characteristics of the RF propagation environment and to localize the users. Localization is performed in terms of absolute coordinates: latitude and longitude, since we focus on 2D locations in this paper.

A key advantage of EZ is that it does not require any prior knowledge of the RF environment, including the location and transmit power of the APs, information that is often not readily available in settings such as malls and multi-tenant office buildings, where APs have been deployed by many different entities. This is a key advance over prior work on reducing the calibration effort needed for indoor localization [11, 17]. Another key advantage is that EZ does not require any explicit user participation to aid the localization process. In particular, users are *not* required to indicate their current locations, even during the training phase. Finally, in contrast to work on collaborative (ad-hoc) localization [20, 7, 23], EZ only requires measurements of the APs by the mobile node and does *not* require any distance measurements between mobile nodes. So even a single mobile node that traverses the space of interest over time could generate sufficient data for EZ localization, as in the experiments we report here. In contrast, prior work on collaborative localization requires the simultaneous presence of a sufficient number of participating nodes, a requirement that makes bootstrapping non-trivial thereby impeding easy deployment.

We believe that by not requiring any pre-deployment effort or explicit user participation, EZ has the potential of enabling practical and viable indoor localization. For instance, an EZ server in the cloud could automatically construct an RF model for, and thereby

MobiCom'10, September 20–24, 2010, Chicago, Illinois, USA.

enable localization in, an indoor space in any part of the world, based just on the measurements reported by the EZ clients in the space of interest. Such automated operation brings forth a number of novel challenges, which we address here, including filtering the very large number of measurement reports to identify a small and useful subset, and efficiently computing the RF model despite the very large space of possibilities.

The advantage of no pre-deployment effort provided by EZ inevitably comes at the cost of some loss of accuracy relative to localization approaches such as RADAR [4] and Horus [29] that rely on extensive measurement to map the RF environment. However, our experience from deploying EZ in two different office buildings is promising. For instance, in one of the buildings, the median localization error with EZ is 2m, which is somewhat worse than the error of 0.7m and 1.3m, respectively, with Horus and RADAR. On the other hand, EZ's approach of inferring an RF *model* is more robust than that of constructing an RF *map* as in Horus and RADAR. For instance, in the above building when measurements are made using a laptop but then are used to localize a different device — a smartphone — EZ's error remains unchanged at 2m whereas that for both Horus and RADAR degrades to 3m or worse.

2. RELATED WORK

Indoor localization has been an active area of research for the past two decades, initially in the context of robot navigation and more recently in the context of pervasive and mobile computing. Here we provide a brief overview of some key research contributions to this area.

Schemes that require specialized infrastructure : The earliest schemes relied on deploying specialized infrastructure to enable indoor localization. For example, Active Badge [26], uses infrared (IR) beacons and receivers to perform localization. Cricket [22] and Bat [27] rely on ultrasound devices being deployed at various locations within the indoor environment as well as on the mobile devices. Recently, RFID based systems such as LANDMARC [19] also have been proposed. *The practical deployment of these systems is hindered by the significant cost and effort involved.*

Schemes that build RF signal maps: The proliferation of static, radio-frequency transmitters such as WiFi APs and GSM towers has enabled localization without the need for additional infrastructure. The basic approach is to fingerprint each location in the space of interest with a vector of received signal strength (RSS) measurements of the various transmitters. A mobile device is then localized by matching the observed RSS readings against this database. An early system that used this approach with pre-WiFi WLANs was RADAR [4], which used a deterministic fingerprint for each location. Since then several schemes have improved upon RADAR, most notably Horus [29], which employs a stochastic description of the RSS map and uses a maximum likelihood based approach. Commercial localization products have also been built using these methods [8]. Otsason*et al.* [25] has demonstrated that GSM signal strength from various towers can also be used for indoor localization. SurroundSense [2] builds a map using several features found in typical indoor spaces such as ambient sound, light, color, etc., in addition to WiFi RSS. *All these schemes, however, entail a considerable amount manual effort to perform detailed measurements across the entire indoor space and maintain the RF map over time.*

Efforts have been made to reduce the mapping effort, for instance, by performing measurement at a coarser, room-level granularity [12]. However, the overall pre-deployment effort remains substantial. DAIR [3] eliminates the need for mapping, but it assumes a very dense deployment of WiFi transmitters, much denser than typical WiFi deployments.

Model-Based Techniques : An RF propagation model (*e.g.*,the log-distance path loss (LDPL) model) can be used to predict RSS at various locations in the indoor environment. The advantage of using these models is that it reduces the number of RSS measurements dramatically compared to RF fingerprinting schemes, albeit at the cost of decreased localization accuracy. Since RF propagation characteristics vary widely, the model parameters would have to be estimated specifically for each indoor space in question.

TIX [11] assumes that the transmit power and locations of all WiFi APs is known. The APs are modified to measure the RSS of the beacons from neighboring APs. Linear interpolation is then used to estimate the RSS at every location in the indoor space, which is then used for localization. To allow unmodified, off-the-shelf APs to be used, Lim *et al.* [17] employ WiFi sniffers at known locations. These sniffers measure the RSS from the various APs and use the LDPL model to construct an RSS map. ARIADNE [13] also deploys sniffers at known locations but makes use of a more sophisticated ray-tracing model based on detailed floor maps and uses simulated annealing to estimate radio propagation parameters. Finally, Madigan *et al.* [18] use a Bayesian hierarchical approach for indoor localization, which avoids the need to know the locations of the training points. However, they still depend on knowledge of the AP locations, besides assuming that the path loss exponent in the LDPL model is the same for all APs. *While these methods cut down the measurement effort, they still require effort in terms of placing infrastructure such as sniffers, extending the capabilities of off-the-shelf APs, and obtaining information on the floor plans, or at least knowledge of AP placement and power settings.*

Localization in Indoor Robotics : For a robot to navigate through an indoor environment, it must have the ability to determine its current location. Initial approaches provisioned the robot with a map of the indoor environment, allowing it to determine its location by comparing its observed environment (using ultra-sound, LADAR sensors, etc.) to the map. A significant step in the area of robotics was Simultaneous Localization and Mapping (SLAM) [16], which allowed a robot to build a map of the indoor environment (in terms of walls and other obstructions) while simultaneously determining its location with respect to the constructed map. WiFi-SLAM [9] extends this to building a WiFi RSS map using a mobile robot. The robot uses its onboard odometer to determine the distance it has moved between measurement points. Knowing a few of these locations (using GPS or certain landmarks) allows estimating the parameters of the LDPL model. *In contrast, EZ builds an RF model without the benefit of sensors such as odometers and LADARs.*

Ad-Hoc localization: Finally, there has been work on ad-hoc localization, wherein a set of nodes, some of which may know their own locations (i.e., landmark or anchor nodes), collaborate to enable all nodes to locate themselves. The earliest notable examples are DV-Hop and DV-Dist by Nicelescu *et al.* [20], SPA by Capkun *et al.*[7] and N-Hop multilateration by Savvides *et al.* [23]. These scheme assume that the nodes can estimate the distance between each other, while other schemes use angle of arrival information [21]. There also exist more approximate methods such as [14], which uses the connectivity graph among the nodes to perform localization in outdoor environments. The target environment was outdoors and it is unclear how well a simple connectivity based approach would carry over to indoor environments. Likewise, Sextant [10] uses a geometric approach that may not carry over to indoor environments. *All of these approaches assume node-to-node communication, which requires the simultaneous presence of multiple nodes in the space of interest. This requirement makes bootstrapping in such localization procedure difficult.* While the presence of fixed nodes such as APs might alleviate this problem in

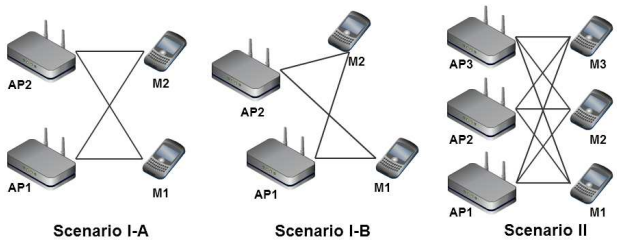


Figure 1: Knowing enough distances between APs and mobile devices allows unique determination of their relative locations

principle, the difficulty on practice is the need for the APs to be modified to participate in the ad-hoc localization protocol.

EZ compared to prior work: EZ works with existing WiFi infrastructure, requiring neither modifications to the APs nor deployment of additional infrastructure. It does not require knowledge of the floorplan, AP placement or power settings. Again, it does not require explicit mapping of the indoor space, whether manually or using a robot. Finally, EZ only relies on measurement of the APs' signal at the mobile node(s), not on any mobile-to-mobile measurement. Hence EZ could even work with measurements by a single device which, over time, traverses the space of interest.

3. LOCALIZATION USING EZ

To provide a physical intuition to the working of EZ, we start with the example depicted in Figure 1. In scenario I-A, two mobile nodes (M1 and M2) have measured their distances (d_{11},d_{12},d_{21},d_{22}) from two APs (AP1 and AP2) with the hope of determining their locations relative to the two APs. This is however not possible, since the same set of distances allows for several different possibilities for relative locations (an alternative is depicted in Scenario I-B). For a set of three APs and three mobile user locations (as depicted in Scenario-II), it can be shown that knowing all nine pairs of distances between APs and mobile users allows for only one possible realization for their relative positions. Such a structure is deemed *localizable*, (or *globally rigid*) *i.e.*, the entire set of locations can be translated, rotated and reflected (flipped) but not distorted in any manner if all distances are to be preserved. Localizability is a well studied area and conditions for localizability have been well studied [28]. In general *given "enough" distance constraints between APs and mobile devices, it is possible to establish all their locations in a relative sense. Knowing the absolute locations of any three non-collinear mobile devices then allows determination of the absolute locations of the rest.*

In practice, however, the distances between mobile devices and APs can only be inferred from RSS values.

$$p_{ij} = P_i - 10\gamma_i \log d_{ij} + \mathbf{R} \tag{1}$$

$$d_{ij} = \sqrt{(\mathbf{x_j} - \mathbf{c}_i)^T (\mathbf{x_j} - \mathbf{c}_i)} \tag{2}$$

In Eqn 1, the j^{th} mobile user located at a distance d_{ij} (measured in meters) from the i^{th} AP sees a signal strength of p_{ij} (measured in dBm). The location of the i^{th} AP and the j^{th} mobile user measurement are represented by 2D vectors $\mathbf{c}_i$ and $\mathbf{x}_j$ respectively in Eqn 2. P_i is the RSS from the i^{th} AP at a distance of one meter (referred to as *transmit power* henceforth). The path loss exponent γ_i captures the rate of fall of RSS in the vicinity of the i^{th} AP. The higher the value of γ_i, the steeper is the fall of RSS with distance. The need for having a different γ_i for each AP arises from the fact that rate at which RSS falls with distance depends on the local environment. RSS from an AP that is located in an area surrounded by walls, people and other obstacles might decay at a much faster rate compared to the same from other APs in the indoor environment that enjoy relatively freer signal propagation. $\mathbf{R}$ in Eqn 1 is a random variable that hopes to capture the variations in the RSS due to multi-path effects, asymmetries in the physical environment (*e.g.,*obstructions) and other imperfections in the model itself.

Based on the LDPL model d_{ij} can be computed as,

$$d_{ij} = 10^{\left(\frac{P_i - p_{ij}}{10\gamma_i}\right)}. \tag{3}$$

Eqn 3 assumes the *a priori* knowledge of P_i and γ_i. EZ, takes a novel approach to estimating these. Given a set of RSS observations between APs and mobile users (p_{ij}), EZ treats P_i and γ_i as unknowns in addition to the unknown locations of APs and mobile users. It then solves the set of simultaneous equations formed by the LDPL model for each RSS observation.

Assume that there are m APs and n unknown locations on a floor. For simplicity assume that all the m APs are visible from each of the n locations (this assumption will be relaxed later in this section). The total number of RSS observations and hence the total number of LDPL equations will be mn. Assuming 2D locations, each of the n locations has two unknowns namely the x and y coordinates (in $\mathbf{x}_j$ in Eqn 2). Each of the m APs has four unknowns namely P_i, γ_i and its 2D location. The total number of unknowns is thus $4m + 2n$.

While mn grows in a quadratic fashion, $4m + 2n$ grows linearly. This suggests that given enough locations (such that $mn > 4m + 2n$), there will be eventually enough constraints in the system of equations to make the system uniquely solvable. Closer examination however, reveals that the system of LDPL equations is scale, translation, rotation and reflection invariant (we do not include the proof due to space constraints). *In other words, a solution to LDPL equations will yield locations that are a scaled, translated, rotated and/or reflected version of the true locations.* Knowing three true, non-collinear locations (either AP or mobile users) then, all the other true locations can be determined. In our implementation, these true locations are obtained opportunistically, when GPS enabled mobile devices gain access to GPS at the edges of the indoor environment such as entrances and near windows.

The above description of relative localization followed by anchoring in an absolute coordinate space, makes for a clean, conceptual separation between the two steps. However, in our implementation, we found it advantageous to combine the two steps, by directly using the absolute locations (obtained opportunistically through GPS) throughout the solution procedure, as we elaborate in Section 4.2.

3.1 The Nature of LDPL Equations

LDPL equations (Eqn 1) are a system of simultaneous non-linear equations. To the best of our knowledge there exists neither an analytical solution nor any prior work that analyzes them. In this section we attempt to provide the reader with some crucial insights into the nature of solutions to these equations.

3.1.1 Solving for the Parameters of a Solitary AP

It is well known that distances from at least three known non-collinear locations are necessary to uniquely determine an unknown location (using trilateration). A corresponding question in EZ is, *what is the minimum number of known locations at which RSS measurements must be taken in order to uniquely determine the four AP parameters namely P,γ and the 2D location?*

If P_i and γ_i are known, then an RSS measurement p_{ij} can be

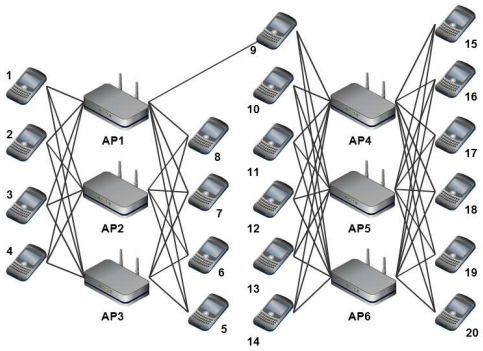


Figure 2: Illustrating non-localizability

converted into the distance d_{ij} between the unknown location ($\mathbf{x}_j$) and the i^{th} AP using the equation Eqn 3. A minimum of three such RSS measurements at known locations are then required to uniquely establish the location of the AP. In the absence of the knowledge of P and γ, then, two additional measurements (constraints) will be required to uniquely determine their values. In other words *five RSS measurements are required to uniquely determine an AP*. It also follows that *given RSS measurements from only four known locations, there will be two possible solutions for (P, γ and location) that satisfy the set of four RSS measurements*. Due to the lack of space we do not present a formal proof.

3.1.2 The Notion of Co-Circular Dependency

It is well known that three or more collinear locations cannot be used in trilateration to determine an unknown location. A similar yet slightly different situation arises in solving LDPL equations when all the locations where RSS observations were taken are co-circular with respect to the AP *i.e.,* lie on a circle centered around the AP's location. In this special case, while the location of the AP can be ascertained as the center of the circle passing through these locations, it is impossible to uniquely determine both P and γ. *In EZ thus, one must avoid observations that have almost the same RSS values from the same AP.*

3.2 Localizability

While a necessary criterion for the existence of a unique solution to the system of simultaneous equations is that the number of equations should greater than or equal to the number of variables, this is by no means a sufficient condition. In Figure 2, for example, there are 6 APs (AP_1 through AP_6) and 20 mobile user locations from which RSS observations were made. An edge between an AP and a mobile user location is drawn iff the AP can be seen from that location. The number of LDPL equations in this system is 61, and the number of variables is 58. However, from the very structure it is clear that the system is not localizable since the group of APs, $AP1$ through $AP3$ is free to rotate about $AP4$ through $AP6$.

The localizability question in EZ is as follows: *given a set of RSS measurements at some unknown and known locations, is it possible to determine a unique set of coordinates for all the unknown locations by solving the corresponding EZ equations?* The question is extremely relevant because, in practice not all APs may be visible from all locations in the indoor environment. Unfortunately, determining the necessary and sufficient conditions under which a set of LDPL equations has a unique solution is still an open problem that we hope to tackle in the future.

However, for most practical scenarios, it is possible to determine whether or not a system of LDPL equations can be uniquely solved by making sure that following three conditions are satisfied.

C1 : Each unknown location must see at least 3 APs.
C2 : Each AP must be seen from at least 5 locations (known or unknown).
C3 : The Jacobian of the system of LDPL equations must have a full rank (equal to the number of variables) for a random choice of the AP parameters and unknown locations.

Conditions C1 and C2 follow from the discussion in Section 3.1. Condition C3 essentially linearizes the system of LDPL equations into the form $\mathbf{Jy} = \mathbf{k}$, where $\mathbf{J}$ is the Jacobian, $\mathbf{y}$ is a vector containing all unknown parameters and $\mathbf{k}$ a constant vector. The lack of full rank then exposes any insufficient coupling in the equations *i.e.,* situations similar to that depicted in Figure 2.

4. COMPUTATIONAL CHALLENGES

While there are several different approaches to solving a set of over-determined equations, in our implementation, we attempt to find a solution that minimizes the least mean absolute error,

$$J_{EZ} = \frac{1}{N} \sum_{ij} \left| P_{ij} - P_i^0 + 10 * \gamma_i \log d_{ij} \right| \tag{4}$$

In Eqn 4 , N is the total number of EZ equations.

J_{EZ} is a non-linear objective function and to the best of our knowledge, does not allow for an analytical closed form solution. Optimization schemes such as the Newton Raphson Method [24] or Gradient Descent [1] (GD) are iterative schemes that start from an initial guess and find the closest local minimum. In our initial trials we found that such schemes fail to find a solution to J_{EZ} since the number of local minima in J_{EZ} is immense. The other alternatives are search techniques such as simulated annealing [15] or genetic algorithms [6] (GAs). While genetic algorithms can search the solution space efficiently, they can miss local minima that might provide a reasonably good solution. Consequently, to obtain the benefits of both these approaches, in our implementation, we used a hybrid approach that used gradient descent to refine the solutions generated by a GA.

Solving LDPL equations using the GA can take a few minutes to several hours depending on the size of the problem. However, these equations need only be solved once (or periodically once every a few days to refresh the model) to determine the AP locations, their transmit powers and the path loss exponents. Once having estimated the model, new location queries can be answered through standard techniques such as trilateration (by converting RSS measurements to distances Eqn 3) in real-time.

4.1 The Genetic Algorithm

The GA starts by picking an initial set of solutions (*initial generation*) randomly and refining them using gradient descent. A solution consists of a vector of values of all the unknowns to be solved in the LDPL equations. The fitness of each solution is then evaluated by computing $\frac{1}{J_{EZ}}$ for the solution. Thereafter, consecutive generations of solutions are generated in the following manner:
1. 10% of the solutions with the highest fitness are retained.
2. 10% of the solutions are randomly generated.
3. 60% of the solutions are generated by picking two solutions $\mathbf{S}_1^{old}$,$\mathbf{S}_2^{old}$ from the previous generation (parent solutions) and mixing them using a random convex linear combination. In other words

$$\mathbf{S}^{new} = \mathbf{a} \bullet \mathbf{S}_1^{old} + (\mathbf{1} - \mathbf{a}) \bullet \mathbf{S}_2^{old} \tag{5}$$

Here, $\mathbf{a}$ is a random vector with each element independently randomly drawn from $(0, 1)$, $\mathbf{1}$ is vector with all its elements equal to 1 and $\bullet$ represents a vector dot product. The newly generated solution is then refined using GD.

4. The remaining 20% solutions are generated by randomly picking a solution from the previous generation and perturbing it (*perturbation based mutation*) by adding (or subtracting) random values (drawn from an exponential distribution to allow occasional large perturbations) to all the locations, P_i and γ_i. The solutions is then refined using the GD.
As generations evolve, solutions with higher fitness are discovered. The GA terminates when solutions do not improve for ten consecutive generations.

4.2 Reducing the Search Space

When the solution space is extremely large, a randomly picked solution is likely to be far from the optimal solution. Consequently it may take a large amount of time before the GA stumbles upon it. Narrowing the search space can dramatically reduce running times. The most obvious way to narrow search space is to limit the search space of the variables. For example, knowing the dimensions of the floor one can limit the search of the locations to within the floor perimeter. For AP transmission powers we chose a generous search space namely $(-50, 0)dBm$. For γ_i we chose the search space $(1.5, 6.0)$. We believe that these ranges will be accommodating for most practical indoor deployments.

Another way to narrow the search space is to leverage constraints inherent to the problem. Given m APs and n locations, as discussed in Section 3 there are a total of $4m+2n$ variables to be picked randomly. However, having picked all the $4m$ AP parameters the $2n$ unknown locations can be determined through trilateration (using Eqn 3 to convert RSS to distances). Thus, the GA needs to randomly pick only $4m$ unknowns rather than $4m + 2n$. The search space reduces exponentially with each eliminated variable.

The search space can be further reduced by using the already determined (or known) locations. For example, suppose that a particular AP can be seen from three known locations. Then after randomly picking P and γ, its location can be uniquely determined. In general, the underlying constraints in the LDPL equations can be listed as follows:
R1 : If an AP can be seen from five or more fixed (or determined) locations, then all four of its parameters can be uniquely solved.
R2 : If an AP can be seen from four fixed locations, there exist only two possible solutions for the four parameters of the AP.
R3 : If an AP (say i^{th} AP) can be seen from three fixed locations. Then after picking γ_i randomly from (1.5,6.0) there exist only two possible solutions that satisfy the observations for the AP location and its transmission power.
R4 : If an AP can be seen from two fixed locations, then, having picked P_i and γ_i randomly, there will be only two possible solutions for its location.
R5 : If an AP can be seen from one location only, then, after picking P_i and γ_i randomly, the AP can only lie on a circle of radius given by Eqn 3 centered about the known location.
R6 : If the parameters for three (or more) APs have been fixed, then all unknown locations that see all these APs can be exactly determined using trilateration.
Constraints R1-R6 indicate that after selecting only a few of the variables randomly, the rest can be deterministically computed.

Based on these underlying constraints in EPL equations, we devised the *EZ Random Solution Generation Algorithm* (ERSGA). ERSGA attempts to minimize the number of variables that need to randomly picked among the entire set of variables in a given set of EZ equations in a greedy fashion. The essential idea behind ERSGA is to start by determining AP parameters with as many known locations as possible. Upon determining AP parameters for three or more APs, locations that can see these APs can be determined using trilateration. These determined locations in turn can be used to determine the parameters of some other APs. The procedure continues until all APs and locations have been determined. The pseudo code for ERSGA is provided below:

```
 1: Function ERSGA(L_done, C_done, O, l, base)
 2: repeat
 3:    change = false
 4:    for i = 1 to m do
 5:       if i ∉ C then
 6:          set O = p_ij | j ∈ L_done
 7:          if |O| ≥ l then
 8:             {c_i, P_i, γ_i} = APRandomInit(l,O,L_done)
 9:             C_done = C_done ∪ {i}
10:             change = true
11:             if l < base then
12:                return true
13:             end if
14:          end if
15:       end if
16:    end for
17:    if change = false then
18:       if l > 0 then
19:          change = ERSGA(L_done, C_done, O, l − 1, base)
20:          if l < base then
21:             Return change
22:          end if
23:       else
24:          Return change
25:       end if
26:    end if
27:    for j = 1 to n do
28:       if j ∉ L then
29:          set O = p_ij | i ∈ C_done
30:          if |O| ≥ 3 then
31:             x_j = Trilaterate(O,C)
32:             L_done = L_done ∪ {j}
33:             change = true
34:          end if
35:       end if
36:    end for
37: until change = true
38: Return change
```

ERSGA is a recursive algorithm that takes five inputs. L_{done} the set indices of all locations where RSS observations were taken that have been determined so far. C_{done} the set of indices of APs with their AP parameters determined. O is the set of RSS observations $p_{1\cdots n,1\cdots m}$. l is the recursion level which searches for the constraint $\mathbf{R}i$. $base$ takes a value of 5 if there are 5 or more known locations, otherwise it takes the value of the number of known locations. The entire procedure begins by initializing L_{done} with indices of all the known locations, C_{done} as an empty set and $l = base$. The function APRandomInit(l,O,L_{done}) (line 8) finds a set of random AP parameters given l determined (or known) locations based on constraint rule $\mathbf{R}i$. For each value of l in APRandomInit(l,O,L_{done}), a different strategy is used to determine the random AP parameters based on constraints R2-R6. For example, in case of R2 or R3 ($l = 4, 5$), the values of the AP parameters are determined through an exhaustive search over several combinations of P and γ in combination with trilateration. In case of R4 ($l = 3$), γ is chosen randomly and P is searched exhaustively to determine local minima in

Mobile Device	RSS (in dBm)
Laptop Xenovo X61	-41
HP IPAQ #1	-43
HP IPAQ #2	-31
Samsung SGHi780 #1	-51
Samsung SGHi780 #2	-49
HTC ADV7510	-49
HTC ADV7501	-37

Table 1: Difference in RSS Readings across Mobile Devices

the mean absolute error. At $l = 0$, all AP parameters are generated randomly.

5. ACCOMMODATING RECEIVER GAIN DIFFERENCES

While ideally all mobile devices should measure the same RSS at the same location, they often do not. Table 1 shows the average RSS measured by different mobile devices simultaneously and at the same location, with line-of-sight to an AP. Such differences can arise from differences in the receiver gains of these mobile devices and from calibration offsets. Since several different mobile devices can participate in EZ, such differences can potentially increase localization errors unless these are compensated for. Haeberlen *et al.* [12] have suggested maintaining a database of pre-measured differences in receiver gains among various, widely-used mobile devices. However, in our experiments we have found that there are gain differences of 2-7dB even among devices of the same make and model. To account for these gain differences we introduce an additional unknown parameter G, the *receiver gain* for each user. In other words, the LDPL model becomes:

$$p_{ij}^k = P_i - G^k + 10 * \gamma_i \log d_{ij}^k + \mathbf{R} \tag{6}$$

Here, k indicates data specific to the k^{th} mobile device, so G^k is the receiver gain of the k^{th} mobile device. G^k is then estimated using the genetic algorithm along with all the other parameters.

As discussed in Section 4, narrowing the search space for the parameters of the LDPL model can provide significant gains in the execution time and performance of the genetic algorithm based solver. A span of $(-20, 20)$ dB provides a generous range to search for differences in receiver gains. However, searching for the gains of several mobile devices even in this limited range is not easy. Hence, in our implementation, we use a novel scheme — the Relative Gain Estimation Algorithm (RGEA) — to provide a coarse-grained estimate of the gain differences among various mobile devices. In other words, RGEA estimates the difference in gain, $\Delta G^{ij} = G^i - G^j$, between the i^{th} and the j^{th} mobile devices. In addition to estimating ΔG^{ij}, RGEA also estimates the uncertainty, $\sigma(\Delta G^{ij})$, in the estimate of ΔG^{ij}. This information helps the GA significantly narrow the search space for the receiver gains of the mobile devices.

5.1 Relative Gain Estimation Algorithm

The difference in RSS measurements obtained using two different devices at the same physical location (or locations that are "close") will be equal to the difference in the devices' receiver gains. Most RSS measurements in EZ are collected at unknown locations. How then do we determine that RSS measurements from two devices were taken at the same or proximate locations? To overcome this challenge, EZ uses the implication from Equation 6 that the difference in RSS from two APs measured by a given device at a location, $(p_{i_1 j}^k - p_{i_2 j}^k)$, is independent of its receiver gain.

Suppose that two mobile devices, k_1 and k_2, took RSS measurements from m APs of $Q_{j_1}^{k_1} =< p_{1j_1}^{k_1}, p_{2j_1}^{k_1}, \cdots, p_{mj_1}^{k_1} >$ and $Q_{j_2}^{k_2} =< p_{1j_2}^{k_2}, p_{2j_2}^{k_2}, \cdots p_{mj_2}^{k_2} >$ from two (unknown) locations, j_1 and j_2. To factor out the (unknown) receiver gain, we subtract the devices' respective RSS measurements corresponding to the first AP from the RSS of each of the remaining APs. This yields the vectors, $V_{j_1}^{k_1} =< 0, p_{2j_1}^{k_1} - p_{1j_1}^{k_1}, \cdots, p_{mj_1}^{k_1} - p_{1j_1}^{k_1} >$ and $V_{j_2}^{k_2} =< 0, p_{2j_2}^{k_2} - p_{1j_2}^{k_2}, \cdots, p_{mj_2}^{k_2} - p_{1j_2}^{k_2} >$. If $V_{j_1}^{k_1}$ and $V_{j_2}^{k_2}$ are "close" to each other, it means that the RSS measurements, $Q_{j_1}^{k_1}$ and $Q_{j_2}^{k_2}$, are similar modulo an offset. Hence, it is very likely that the locations, j_1 and j_2, where these measurements were made are in proximity. In our implementation, if the average difference between the elements of $V_{j_1}^{k_1}$ and $V_{j_2}^{k_2}$ is less than 3dB, then locations j_1 and j_2 are deemed as being proximate. RGEA uses this criterion to create a set, $M^{k_1 k_2}$, of pairs of measurements, $(p_{ij_1}^{k_1}, p_{ij_2}^{k_2})$ such that j_1^{th} and j_2^{th} locations are proximate. Then,

$$\Delta G^{k_1 k_2} = \frac{1}{|M^{k_1 k_2}|} \sum_{(p_1, p_2) \in M^{k_1 k_2}} (p_1 - p_2) \tag{7}$$

$$\sigma\left(\Delta G^{ij}\right) = \frac{1}{|M^{k_1 k_2}|} \sqrt{\sum_{(p_1, p_2) \in M^{k_1 k_2}} \left(p_1 - p_2 - \Delta G^{k_1 k_2}\right)^2} \tag{8}$$

In this manner RGEA computes ΔG^{ij} and $\sigma(\Delta G^{ij})$ for all pairs of mobile devices whenever possible. Note that in many cases two mobile devices i and k might not have even a single pair of measurements from proximate locations. Nevertheless, knowing ΔG^{ij} and ΔG^{jk}, ΔG^{ik} can be estimated transitively as $\Delta G^{ik} = \Delta G^{ij} + \Delta G^{jk}$. Hence, we can determine ΔG^{ij} even for devices that have no measurement locations in common (or in proximity) using the additive property of the receiver gains.

To effect the transitive estimation noted above, RGEA constructs a graph with a node assigned to each mobile device. An edge is drawn between two nodes if and only if there was at least one pair of measurements that came from proximate locations. Each connected component in this graph, then, represents the set of devices whose gain differences can be estimated relative to each other. For each such component, RGEA picks a node (device) randomly as the root node and assigns it a gain by sampling uniformly randomly in the interval $(-20, 20)$ dB. To estimate gains corresponding to the other nodes (devices) in the connected component, RGEA starts with equations of the form:

$$G^j - G^i = \Delta G^{ij} \tag{9}$$

RGEA then estimates all the gains, G^k (relative to the gain that was randomly assigned to the root node), by solving the simultaneous system of equations 9 in a weighted least mean square sense. The weight for each equation is set as the estimated standard deviation, $\sigma(\Delta G^{ij})$, of the gain difference. In our evaluation we found that RGEA estimates receiver gain differences accurately within 1-3dB.

5.2 Localizing New Device with Unknown Gain

After the locations, transmit powers and path loss exponents for the APs have been estimated, EZ uses these to localize new mobile devices in real time. In the absence of receiver gain differences, as discussed in Section 3, the measurement from each AP is converted into an estimate of the distance from that AP using Equation 3. Knowing at least three such distances from APs, we can use standard trilateration to estimate the location of the mobile device. However, this approach will fail when there are differences in receiver gain. Hence, to localize a new mobile device, EZ treats its

gain also as an unknown. Then, EZ constructs the set of simultaneous, gain-independent equations as:

$$p^k_{i_2 j} - p^k_{i_1 j} = P_{i_2} - P_{i_1} + \gamma_{i1} \log(d_{i_1 j}) - \gamma_{i2} \log(d_{i_2 j}) \quad (10)$$

In Equation 10, d_{ij} is the distance of the i^{th} AP from the unknown location of the mobile device. The unknown location of the new mobile device is then estimated by solving these set of equations in a least mean squared sense. To the best of our knowledge, no analytical solution exists to solve the above set of simultaneous equations (since the distances noted in Equation 10 embed the unknown coordinates of the mobile device in quadratic form). Hence, EZ finds the solutions by searching in a bounding box around the APs.

6. IMPLEMENTATION OF EZ SYSTEM

The EZ system has been built based on a client-server architecture. Mobile devices (laptops, cell phones, netbooks *etc.*) act as *EZ Clients* and connect to a *EZ Server*. The EZ server is responsible for providing the location information to the EZ clients. The current system can support multiple mobile devices to communicate with the server and obtain location information. All communication between the client and the server has been implemented over TCP sockets. While the current implementation uses WiFi for communication, extensions to use a cellular interfaces are possible.

6.1 EZ Clients

The EZ client is a piece of software that can be installed on a variety of mobile devices. We have implemented the EZ client for Windows Mobile 6 and tested it on a variety of smartphones such as HTC Advantage, Samsung SGH i780 and HP iPAQ. The EZ client has two important tasks. First, it provides location information to mobile applications residing on the device and second it assists the EZ server in constructing an RSS map of the indoor environment.

6.1.1 Location Queries

The client first checks to see it can obtain the location from GPS on the device. If no GPS signal is available, it scans its environment for WiFi APs in its view. The scanner scans for a few seconds (3 in our implementation) collecting beacons from each AP it sees. It then transmits a list of the mean and standard deviation of the RSS seen from various APs to the EZ server over the collected data. Using the mean reduces errors due to multi-path. Scans that yield RSS measurements with a standard deviation greater than 10 dB are deemed unreliable and are discarded.

The EZ server then uses the computed model of the RSS environment to determine the location of the mobile device in real time and responds to the EZ client. The current implementation of the EZ client comprises about 2000 lines of C# and C++ code.

6.1.2 Assisting the EZ Server in Model Creation

For new indoor spaces, an RSS model must be first generated before queries can be answered. For this, the EZ clients scan for WiFi APs in their range and transmit the observed mean and standard deviation of RSS to the EZ server exactly in the same manner as a location query. While in our implementation devices periodically perform a scan and push the information to the EZ server, this can be easily implemented as a pull, where the EZ server requests for a scan. Pull is desirable since mobile devices then need scan only when necessary and this can lead to significant energy savings.

6.2 The EZ Server

The EZ server has two important functions. First, it responds to location queries from the EZ clients in real time and second it uses the EZ algorithm to construct and maintain the model for the indoor environment in question.

Since the data from a large number of mobile users can be very large and in many cases not useful for training. Thus, EZ performs pre-filtering and selects only a useful subset of the available data to learn the model (see Section 7). The EZ algorithm (described in Section 4.1) then uses this data to generate the model. The EZ algorithm is computationally intensive and may require several minutes to hours depending on the specifics of the indoor space an the nature of data. The EZ server, may potentially reside in the cloud and leverage its computing resource to construct the RF model.

The GA is inherently amenable to massively parallel computation, since each of the operations such as crossover and mutation can be parallelized. To take advantage of the parallelism, in our implementation, we implemented each such operation as a separate thread. This allowed us to take advantage of multiple cores at the EZ Server. The EZ GA was run on a HP PRoline 8-core server class machine. Further parallelism can be exploited if the GA is implemented using parallel programming paradigms such as using an MPI interface. The current implementation of the EZ server comprises about 7000 lines of C# and Python code.

7. CHALLENGES IN REAL ENVIRONMENTS

In this section we describe in detail the novel practical challenges that we had to address on the path to making a EZ based localization system work in real environments.

7.1 Selecting the Right Set of APs

The sheer number of WiFi APs that could be seen on a given floor in our deployments came to us as a rather unexpected surprise. For example, in one of our deployments we could see a total of about 160 APs across a single office floor! A large fraction of these actually belonged to neighboring office buildings. Another interesting observation was that often each AP was configured with multiple SSIDs and appears as different APs. Clearly, running EZ on all observed APs would constitute an immense computational hardship at the cost of incremental gains. Consequently, an automated AP filtering mechanism had to be designed that would select the most suitable APs. Note that EZ has no information as to if these APs belonged to the indoor environment of interest or not.

Several naïve approaches to AP selection can be designed based on desirable properties such as coverage, low standard deviation in RSS, and high average signal strength. For EZ however, our goal was to minimize the number of selected APs while preserving its performance. To this end, we developed the *APSelect* algorithm. The essential idea behind APSelect is to select each AP to provide information that other selected AP do not. Information theoretically speaking the mutual entropy between the data from any two APs must be high. *APSelect* however, uses a more approximate and simpler approach. It starts by computing a similarity metric for each pair of APs based on observed RSS. APs with the most similar data are then clustered together. A representative AP is then elected from each cluster.

To compute the similarity metric, all the RSS observations p_{ij} from the i^{th} AP at the j^{th} location are first normalized to lie within the range (0,1) by dividing them by 100 (since observed RSS typically lie in the range 0 to -100 dBm) and then their mean is subtracted from each reading to give a normalized RSS observation $p^{normalized}_{ij}$. When RSS readings are not available at certain locations, these gaps are filled with a reading of -100 dBm assuming that the RSS is below the receive threshold of the receiver. The

similarity metric is then computed as,

$$\lambda_{ij} = 1 - \frac{1}{n}\sum_{k} |\hat{p}_{ik} - \hat{p}_{jk}| \quad (11)$$

λ_{ij} essentially measures how similar RSS readings were across all locations (known and unknown) for the two APs.

For clustering similar APs, we used hierarchical clustering. Initially each AP represents a single cluster and at each following step two of the most similar clusters are merged. The similarity between two AP clusters is computed as the average similarity between all inter-cluster pairs of APs. For selecting a representative AP within a cluster all APs are first ranked in the order of the number of known locations that can be see the AP. An AP that can see a larger number of known locations is given a higher priority. Among APs with same priority, an AP which has the highest average similarity to rest of the APs in its cluster is selected as the cluster head. We select the minimum number clusters while ensuring that no two clusters have a similarity greater than 90%. We demonstrate the efficacy of *APSelect* in Section 8.

7.2 Selecting a Subset of Locations

For training EZ, ideally RSS observations must come from several different locations across the indoor environment. For example, data from a single mobile user who actively ventures to different places across the indoor space while his mobile device transmits the RSS information to a EZ server. Alternatively, data from a large number of mobile device users spread out across various locations of the indoor space. Given the large amount of data, then, how should we cull out the "useful" subset of data?

To tackle this problem we developed the *LocSelect* algorithm. It works exactly the same as APSelect, except that we flip the problem by treating the selected APs as locations and vice-versa. In other words each new location is selected such that the RSS information it provides has minimum overlap with other selected locations. We demonstrate the efficacy of *LocSelect* in Section 8.

8. DEPLOYMENT AND RESULTS

In this section, we present a comprehensive overview of our evaluation methodology and experimental results, which attempts to answer the following questions:

- What is the cost in terms of localization error paid by EZ to achieve the ease and freedom from pre-deployment effort compared to fingerprinting based schemes?
- How does EZ compare with a model based scheme that has access to AP and measurement locations?
- How does the performance of EZ improve as more and more measurements become available?
- How does the system fare in localizing new devices which are not used in the training process?
- How robust is performance to using multiple devices, with different receiver gains, to build the RF model?
- How effective are the APSelect and LocSelect algorithms?
- What is the computational cost of RF model estimation?

8.1 Deployment in Two Buildings

We have deployed EZ in two different office buildings. The first, henceforth referred to as *SMALL*, is a typical office floor housing around 30 people (Fig 3a). The floor comprises several obstructions in the form of concrete walls, wooden partitions, and glass/metal doors. The second, henceforth referred to as *LARGE*, is a very large floor used as a call center (Fig 3b). This L-shaped floor also contains several obstructions, including pillars, wooden partitions and concrete walls; and houses a few hundred people.

8.2 Comparisons

We compare the performance of EZ against three schemes:

- **RADAR:** RADAR is an RSS fingerprinting scheme, which involves considerable pre-deployment effort in constructing a database of RSS signatures collected from various known locations within the floor. An incoming signature is then matched against this database. The closest match (or the average of k-nearest matches) is returned as the estimated location. We found that averaging with $k = 5$ performs the best and used it in our evaluations.
- **Horus:** Horus improves on RADAR by maintaining a probability distribution of the observed RSS values at various locations instead of single values. Locations are then determined as an average over a few most likely locations, weighed by their likelihood. As in the original implementation [29], we return a weighted average of the top 6 locations.
- **EZ+Loc:** To evaluate the performance of a model-based scheme when given the benefit of knowing all measurement as well as AP locations, we experimented with an EZ+Loc scheme, which operated as follows. We fed in the locations of all APs and measurement points to EZ, and then estimated the transmit power (P) and path loss exponent (γ) for the APs.

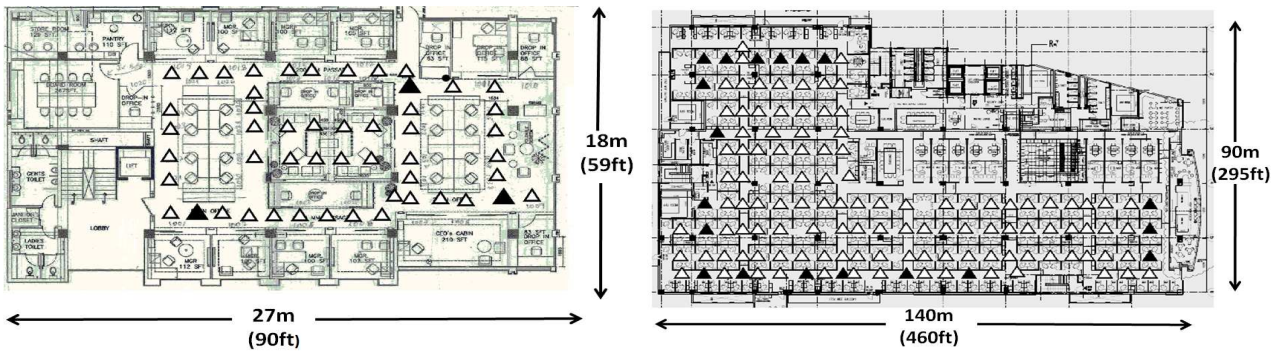
8.3 Experiment Methodology

For our experiments, we used two different kinds of mobile devices: a Lenovo X61 laptop and an HP iPAQ hw6965 smartphone. To build the databases for RADAR, Horus, and EZ+Loc, we collected RSS signatures at grid locations throughout the floor at roughly every 1.5m in *SMALL* and 3m in *LARGE*. At every location, we collected a total of 10,000 beacons, an exercise which took us about 5 minutes per location. For EZ, a user held the mobile device and simply walked across the floor briefly stopping for about 3 seconds at each location to establish ground truth for evaluation. Note that the ground truth information is used only for the evaluation of localization errors, and is *not* supplied to EZ for training. Since our laptop was not equipped with a GPS unit, we used a commercial GPS equipped with a SiRF Star III chipset running Navigon software for obtaining GPS locks. For evaluating the different schemes, we gathered a test data set at a number of locations spread across various sections of the floor. We chose the median and 80^{th} percentile error on this test data set as our evaluation metrics.

8.4 Performance in *SMALL*

In the data collected for EZ, the Lenovo X61 laptop was used to obtain RSS readings from 48 unknown locations (depicted as hollow triangles in Figure 3a), and also 3 known locations (depicted as filled triangles in Figure 3a), where a GPS lock was obtained. Our data showed that 48 different APs (MAC Addresses) were visible from this floor. Many of these APs did not belong to *SMALL*. APSelect selected 4 APs from these 48, which coincidentally all of these belonged to *SMALL*.

For RADAR and Horus we used all the 48 visible APs since this constituted the most information. Some studies [4] have reported that having a larger number of APs sometimes degrades performance. Consequently, we tested RADAR and Horus with smaller subsets of APs obtained from APSelect and found that the best performance (in terms of mean square error) was indeed obtained when all 48 APs were selected. For EZ+Loc, we had location information only for four APs, and we used these APs for localization.



(a) *SMALL* (b) *LARGE*

Figure 3: Floorplans for the buildings.

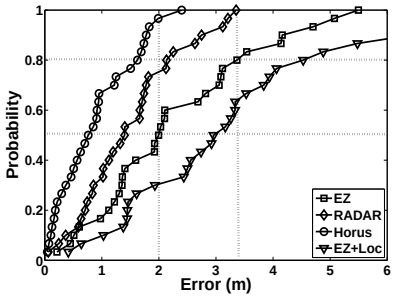


Figure 4: CDF of localization errors in *SMALL* using the Lenovo X61 Laptop.

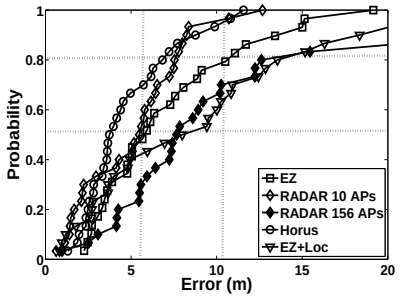


Figure 5: CDF of localization errors in *LARGE* using the Lenovo X61 laptop.

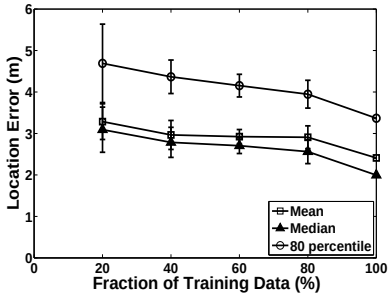


Figure 6: Dependence of EZ's performance on amount of training data

Figure 4 depicts the cumulative distribution of localization errors obtained for all the four localization schemes. Horus performs the best with 50^{th} and 80^{th} percentile error of 0.7m and 1.3m respectively, followed by RADAR with values of 1.3m and 2.1m for the same. EZ provides a 50^{th} percentile error of 2m and an 80^{th} percentile error of 3.3m. An office cube in this floor is roughly 3m in size. Contrary to what we expected, EZ+Loc performed the worst, with error values of 3.1m and 4.4m respectively. To understand the reason for this, we analyzed the RSS map generated by EZ and EZ+Loc and found that EZ's map was closer to reality although the locations and transmit powers of the APs it found did not match exactly with the ground truth. While EZ had the flexibility to compensate for estimation errors in the path loss exponent (γ) by adjusting the estimated locations of the APs, EZ+Loc did not have this luxury. Consequently, it could not compensate for the errors in the model. Thus, fixing the AP locations resulted in degraded performance.

8.5 Performance in *LARGE*

In this experiment RSS data was collected for EZ at 101 different locations within this floor, as depicted by the triangles in Figure 3b. The entire floor is not well ventilated and has few very small windows. As a result we could not obtain a GPS lock at any location within this floor. However, this was an excellent opportunity to test the performance of EZ on a large floor, so we selected points that were closest to the boundaries of the floor and deemed these as known locations. There were 15 such known locations in the training data, depicted as filled triangles in Figure 3b.

We processed the collected data, and found 156 different AP MAC addresses, only a few of which belonged to the office. For EZ+Loc, we obtained locations for 12 of these APs. RADAR and Horus database were created using all the 156 APs. EZ used APSelect algorithm, which picked 10 out of the 156 APs found.

For testing we collected RSS readings at a separate testing set of 40 locations and evaluated all the four schemes over these locations. As in the case of *SMALL*, Horus performed the best with 50 and 80 percentile errors of 4m and 7m, respectively. However, RADAR performed significantly worse (median error of 7m and 80 percentile of 12m), indicating the need for careful selection of APs. In the absence of any specified scheme for the selection of APs for RADAR [4], we used APSelect to pick out a smaller subset of APs. The best performance of RADAR was found with 10 APs, with the 50 and 80 percentile errors being 5m and 7m, respectively. We also ran Horus with different subsets of APs selected using APSelect, but the best performance was achieved with all the 156 APs.

EZ yielded 50 and 80 percentile errors of 7m and 10m, respectively. *This implies that localization accuracy was within two to three cubicles in the call center.* The accuracy is still quite useful since the office building is quite large and houses a few hundred cubicles. As in case of the small building, EZ+Loc performed significantly worse than EZ.

8.6 Dependence on Amount of Training Data

As users cover more ground within the indoor environment and hence more RSS measurements become available for training, we would expect EZ's performance to improve. To evaluate this, we started with measurement data from the full set of 50 locations in *SMALL*. We then selected random subsets comprising 20%, 40%, 60%, 80%, and 100% of the full measurement data, such that each successive subset included the previous subsets. We then trained

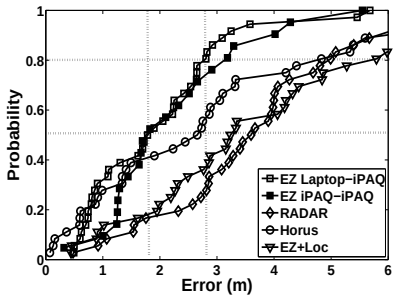


Figure 7: Performance on a new mobile device in *SMALL*

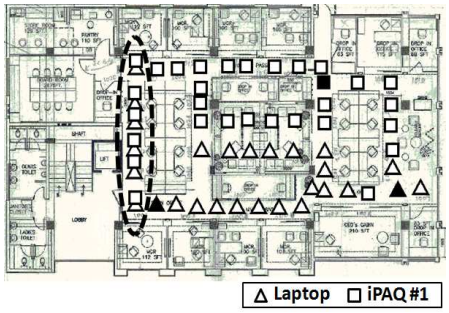


Figure 8: Locations of the two mobile devices used in the EZ model construction

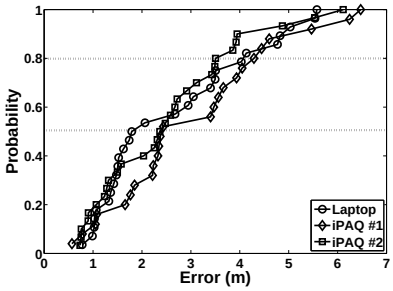


Figure 9: Training with different mobile devices

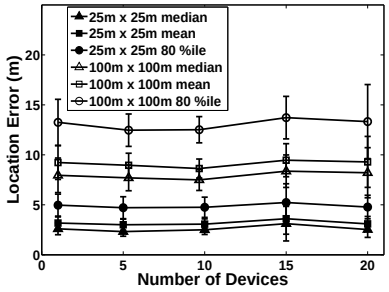


Figure 10: Dependence of EZ's performance on number of different devices

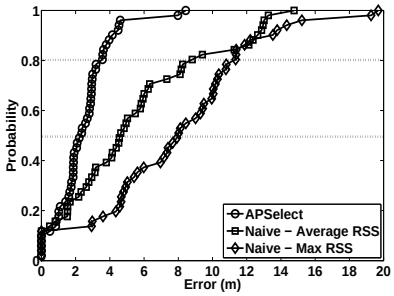


Figure 11: APSelect versus Naive Schemes

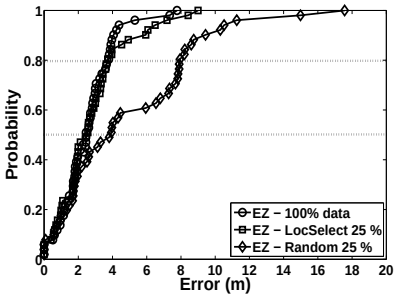


Figure 12: LocSelect versus Random 25%

EZ using these subsets and evaluated localization accuracy using a separate test data set gathered at 30 locations within *SMALL*. We repeated the above procedure of evaluating EZ with random subsets 10 times. Figure 6 depicts the median, mean, and 80^{th} percentile of the localization errors over the 10 runs. Even with only 20% of the training data (i.e., 10 measurement points) being available, EZ is able to achieve a median error of about 3m. As more training data is made available, the median and 80^{th} percentile errors progressively decrease to under 2m and 4m, respectively.

8.7 Performance for a New Mobile Device

How well does EZ localize a new mobile device that has never been used to generate the model? In particular, how does the receiver gain difference among receivers, as discussed in Section 7, affect performance? To answer these questions, we collected RSS signatures at 40 different locations in *SMALL* using the iPAQ smartphone. We then localized these positions using the model generated with measurements made using the laptop. In EZ, in addition to the location of the mobile device, we also estimate its (unknown) receiver gain, G, as discussed in Section 7. We experimentally determined the gain difference to be roughly 11dB, although this information was *not* supplied to EZ. Note that none of the schemes other than EZ provide mechanisms to estimate or correct for errors due to differences in receiver gain.

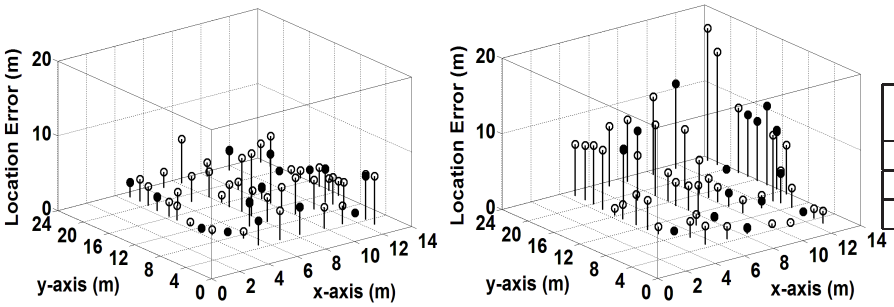
Figure 7 shows the CDF of errors over these 40 points. The first key observation is that EZ outperforms all the other schemes (curve "EZ Laptop-iPAQ" in the figure). This clearly demonstrates the benefit of EZ's estimation and compensation of receiver gain for the new device. EZ yields 50 and 80 percentile errors of 1.8m and 2.8m, respectively. To determine how the performance would have been if the model was built using the same device, we had a user walk around the floor (with intermittent stops lasting a few seconds to collect ground truth information) to train EZ using the iPAQ. The curve labeled "EZ iPAQ-iPAQ" shows the result from this experiment. Indeed it can be seen that the performance is almost similar for the curves "EZ iPAQ-iPAQ" and "EZ Laptop-iPAQ".

The relatively poor performance of (unmodified) RADAR and Horus, as shown in Figure 7, highlights the importance of estimating and compensating for differences in receiver gain, even when multiple devices of the same type are used for measurement. It is likely that Horus and RADAR would perform better if also given the benefit of gain compensation. However, unlike EZ, it is unclear how either Horus or RADAR could automatically estimate the gain for a new device that has not been used to create the RSS map.

8.8 Training with Data from Multiple Devices

All our results with EZ so far have utilized data collected using a single device for training. However, in practical scenarios, multiple users would provide training data. How does EZ perform when multiple devices are used for data collection? To analyze the behavior of EZ under such a setting, we used a mixture of data from the laptop and iPAQ (iPAQ #1) for training. As shown in Figure 8, the laptop user limited themselves to the south side of the building while the iPAQ #1 user restricted themselves to the north side. The two users had a common area of coverage (about 6 locations, indicated by the dotted ellipse) in the region between the kitchen and the entrance to the floor. EZ was able to used these locations to compute the receiver gain offsets between the two devices as discussed in Section 7.

For testing purpose, we collected observations at approximately 30 different locations on three different mobile devices. Two of these were the ones used for training, i.e., the laptop and iPAQ #1. The third was another iPAQ, which we label as iPAQ #2. Experimentally we determined that iPAQ #1 and iPAQ #2 had a receiver



(a) Locations selected by LocSelect (b) Locations selected randomly

Figure 13: Spatial distribution of errors at various locations from EZ.

m	n	Known	Time in minutes	
			Lenovo T61	Server
5	50	3	65	53
5	25	3	38	22
5	12	3	16	12

Figure 14: EZ running times

gain difference of about 7dB while iPAQ-1 and the laptop had a receiver gain difference of 11dB. *Note that this information was not supplied to EZ.* As seen from Figure 9, the performance of all three devices is almost the same, yielding median and 80^{th} percentile errors of about 2m and 4m, respectively.

How does EZ's performance scale as the number and diversity of the mobile devices grows? To answer this question, we built a custom simulator. We assumed two floor sizes: 25m $\times$ 25m and 100m $\times$ 100m. AP locations were generated randomly in the floor and each AP was assigned random values of P (in -20dBm to -50dBm) and γ (in 2 to 5). 5 APs were used for the 25m $\times$ 25m floor and 9 for the 100m $\times$ 100m floor. These choices were based on our experience from actual deployments. Ten batches each of training and testing locations (100 points each) were generated randomly for each floor. From each batch of training locations, five different training data sets were derived, corresponding to there being 1, 5, 10, 15, or 20 devices. A training data set with k devices was derived by creating k partitions in the batch of training locations and assigning each partition to a different mobile device. To generate the "measured" RSS values, our simulator used the LDPL model, with a randomly picked gain (G in Equation 6) in the range (0,20) dB assigned to each device. Temporal variations in RSS were modeled using a Gaussian random variable with a standard deviation of 3dB. Finally, all training locations within 3m from the boundary were assumed to acquire GPS and hence deemed as known locations. Thus, the above procedure ensured that for each batch of locations, we had runs of the experiment that only differed in the number of mobile devices.

To evaluate localization accuracy for a new mobile device, we generated RSS measurements at the locations in the testing set, using another random gain value drawn from (0,20)dB corresponding to the new device. For each training data set, we used EZ to estimate the AP parameters. We then used these to estimate the locations of the new device in the test set and then compute the localization error. Figure 10 depicts the median, mean and 80^{th} percentile errors across the 15 batches of training and testing locations for each of the two floors. We see that the accuracy of EZ does not change significantly with an increasing number of mobile devices. This indicates that the RGEA algorithm (Section 5) is able to effectively estimate and compensate for gain differences across mobile devices.

8.9 The Efficacy of APSelect

We showcase the efficacy of our APSelect algorithm by comparing it with two (naïve) versions of AP selection schemes. Our first scheme, *AvgRSSAPSelect* computes the average RSS seen across all the locations for each AP and selects the strongest APs. The second scheme, *MaxRSSAPSelect* finds the maximum RSS across all locations for each AP and then selects the top APs. To be fair and consistent with APSelect, we selected the same number of APs using the two alternate schemes.

Figure 11 depicts the CDF obtained by training EZ on each of selected set of APs on the *SMALL* data set. We see that the set of APs selected by APSelect significantly outperforms those by the other two schemes. Upon inspecting the distribution of RSS seen at various locations for APs given by AvgRSSAPSelect and MaxRSSAPSelect, we noticed a curious occurrence. Two different pairs of APs selected in both schemes showed almost the same RSS values at all locations. In other words they were co-located APs on different channels. APSelect wisely avoided picking both of them since they were providing the same information.

8.10 The Efficacy of LocSelect

LocSelect carefully picks a subset of the RSS data collected across locations with the goal of reducing model training time while incurring a minimum loss in localization performance. To evaluate the efficacy of LocSelect, we selected 25% of the locations using LocSelect and trained EZ. The estimated AP parameters were then used to estimate error over a separate testing set of 30 locations. To investigate how a randomly picked subset would perform, we then picked 25% of the locations randomly and trained EZ on these randomly picked locations. Localization error was then evaluated over the testing set.

Figure 12 depicts the CDF of the errors obtained from the two strategies. The CDF obtained by running EZ with all the locations is provided as reference. As seen in Figure 12, both the 50 and 80 percentile errors obtained by using just 25% of the unknown locations match those obtained by picking all the unknown locations! Random selection, however, did not perform as well, with the 80 percentile error being almost 7m. Figures 13a and 13b depict the spatial distribution of locations errors across the entire floor. The locations that were picked for training EZ are indicated by solid circles. As seen from Figure 13a and 13b, the distribution of the selected locations appears to be random. However, these locations are, in fact, picked carefully using LocSelect and these help EZ learn the RF model accurately.

8.11 Running Time of EZ

The time taken by EZ to estimate the RF model depends on several factors including the number of unknowns (locations and AP parameters), the nature of data (data that fits the model well is solved faster), the number and placement of known locations,

the distribution of the unknown locations and finally the choice of initial solutions that were randomly picked for the GA and their number. In Table 14 we provide the typical running times that we observed while training EZ. These measurements were conducted by training EZ on two different machines. The first, a Lenovo T61p laptop and the other HP PRoline which was a server class machine with 8 cores. Note that model estimation is an offline task and one that is likely to be repeated infrequently for a given indoor space, so a relatively long running time is not a hindrance.

9. DISCUSSION

There are several challenging extensions to the current state of EZ. In this section we briefly describe a few of these. All of our evaluation has been based on measurements made from within the indoor space of interest. How would we ensure this in practice? The EZ framework from Section 3 does not explicitly assume that the measurements are made within the indoor space of interest. However, measurements taken from outside the space could distort the solution due to signal attenuation from significant obstructions such as the exterior walls. Such RSS measurements would however stand out as outliers, due to their poor fit in Eqn 1, and could be identified and discarded accordingly. We hope to address this in our future work.

Energy is an important consideration for mobile devices. Scanning for WiFi devices or obtaining a GPS lock, and transmitting this information to the EZ server, could consume a significant amount of energy. The energy cost could be reduced by having the EZ server pull in RSS information from mobile devices only when needed. Furthermore, such crowd-sourcing of information can be effected in a manner that balances the burden across mobile devices while also being cognizant of their battery levels. Thus, the energy cost for any individual mobile device would be minimal.

Finally, the parameters of the RF model are likely to be subject to diurnal variations. For instance, RF propagation might be more severely attenuated when an airport or a mall is crowded than when it is sparsely populated. To address this issue, we could construct separate models for different times of the day and borrow from prior work on RF environment profiling [5].

10. CONCLUSION

EZ is a novel, configuration-free indoor localization scheme that uses existing WiFi infrastructure to localize mobile devices. It does not require any pre-deployment effort, infrastructure support, prior knowledge about WiFi APs, or active user participation. EZ learns by collecting data from mobile devices carried by users as they traverse the indoor space of interest in normal course.

We have implemented EZ and compared it against RADAR and Horus, localization schemes that use RF fingerprinting, and also a scheme that leverages knowledge of AP and measurement locations. Based on deployment in two different office buildings, EZ's performance is only somewhat worse than that of schemes that depend on extensive mapping effort (median error of 2m with EZ compared to 0.7m and 1.3m with Horus and RADAR, respectively), while more being robust to device diversity. In future work, we plan to extend EZ to 3 dimensions and, separately, investigate energy efficiency issues.

11. ACKNOWLEDGEMENTS

We thank our shepherd, Alex Snoeren, and the anonymous reviewers for their constructive comments. We also thank Moses Raju, Vinod Mathews and Santhanam Lakshmanan for helping us with conducting experiments in Microsoft GTSC, India.

12. REFERENCES

[1] M. Avriel. *Nonlinear Programming: Analysis and Methods*. Dover Publishing, 2003.

[2] M. Azizyan, I. Constandache, and R. Roy Choudhury. SurroundSense: Mobile Phone Localization via Ambience Fingerprinting. In *MobiCom*, 2009.

[3] P. Bahl, J. Padhye, L. Ravindranath, M. Singh, A. Wolman, and B. Zill. DAIR: A Framework for Managing Enterprise Wireless Networks Using Desktop Infrastructure. In *HotNets*, 2005.

[4] P. Bahl and V. N. Padmanabhan. RADAR: An Inbuilding RF-based User Location and Tracking System. In *INFOCOM*, 2000.

[5] P. Bahl, V. N. Padmanabhan, and A. Balachandran. Enhancements to the RADAR User Location and Tracking System, Feb 2000. Microsoft Research Technical Report MSR-TR-2000-12.

[6] W. Banzhaf, P. Nordin, R. Keller, and F. Francone. *Genetic Programming - An Introduction*. Morgan Kaufmann, 1998.

[7] S. Capkun, M. Hamdi, and J. Hubaux. GPS-Free Positioning in Mobile Ad-Hoc Networks. In *HICSS*, 2001.

[8] Ekahau. http://www.ekahau.com/.

[9] B. Ferris, D. Fox, and N. Lawrence. WiFi SLAM Using Gaussian Process Latent Variable Model. In *JCAI*, January 2007.

[10] S. Guha, R. Murty, and E. G. Sirer. Sextant: A Unified Node and Event Localization Framework Using Non-Convex Constraints. In *MobiHoc*, 2005.

[11] Y. Gwon and R. Jain. Error Characteristics and Calibration-Free Techniques for Wireless LAN-based Location Estimation. In *Mobiwac*, 2004.

[12] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki. Practical Robust Localization over Large-Scale 802.11 Wireless Networks. In *MobiCom*, 2004.

[13] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal. ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System. In *MobiSys*, 2006.

[14] N. B. John, J. Heidemann, and D. Estrin. GPS-Less Low Cost Outdoor Localization For Very Small Devices. *IEEE Personal Communications Magazine*, 7:28–34, 2000.

[15] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science, New Series*, 220:671–680, 1983.

[16] J. Leonard and H. F. Durrant-whyte. Simultanoues Map Building and Localization for an Autonomous Mobile Robot. In *IROS*, pages 1442–1447, 1991.

[17] H. Lim, C. L. Kung, J. C. Hou, and H. Luo. Zero Configuration Robust Indoor Localization: Theory and Experimentation. In *Infocom*, 2006.

[18] D. Madigan, E. E., R. P. Martin, W. Ju, P. Krishnan, and A. Krishnakumar. Bayesian Indoor Positioning Systems. In *Infocom*, 2005.

[19] L. Ni, Y. Liu, C. Yiu, and A. Patil. LANDMARC: Indoor Location Sensing Using Active RFID. In *WINET*, 2004.

[20] D. Niculescu and B. Nath. Ad-Hoc Position System. In *IEEE Globecom*, 2001.

[21] D. Niculescu and B. Nath. Ad-Hoc Positioning System (APS) using AOA. In *Infocom*, 2003.

[22] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *MobiCom*, 2000.

[23] A. Savvides, C. Han, and M. Srivastava. Fine-Grained Localization in Ad-Hoc Networks of Sensors. In *MobiCom*, 2001.

[24] E. Sûli and D. Mayers. *An Introduction to Numerical Analysis*. Cambridge University Press, 2003.

[25] A. Varshavsky, E. Lara, J. Hightower, A. LaMarca, and V. Otsason. GSM Indoor Localization. *Pervasive and Mobile Computing*, 2007.

[26] R. Want and et al. The Active Badge Location System. *ACM Transactions on Information Systems*, Jan 1992.

[27] A. Ward, A. Jones, and A. Hopper. A New Location Technique for the Active Office. *IEEE Per. Comm.*, 4(5):42–47, 1997.

[28] Z. Yang, Y. Liu, and X.-Y. Li. Beyond Trilateration : On the Localizability of Wireless Ad-Hoc Networks. In *InfoCom*, 2009.

[29] M. Youssef and A. Agrawala. The Horus WLAN Location Determination System. In *MobiSys*, 2005.