Mobile Netw Appl
DOI 10.1007/s11036-008-0139-0

Unsupervised Learning for Solving RSS Hardware Variance Problem in WiFi Localization

Arvin Wen Tsui · Yu-Hsiang Chuang · Hao-Hua Chu

Abstract Hardware variance can significantly degrade the positional accuracy of RSS-based WiFi localization systems. Although manual adjustment can reduce positional error, this solution is not scalable as the number of new WiFi devices increases. We propose an unsupervised learning method to automatically solve the hardware variance problem in WiFi localization. This method was designed and implemented in a working WiFi positioning system and evaluated using different WiFi devices with diverse RSS signal patterns. Experimental results demonstrate that the proposed learning method improves positional accuracy within 100 s of learning time.

Keywords localization systems · Wi-Fi network · unsupervised learning · Wi-Fi device variance

A. W. Tsui
Graduate Institute of Networking and Multimedia,
National Taiwan University,
No.1, Roosevelt Road, Sec.4,
Taipei 106, Taiwan
e-mail: arvin@itri.org.tw

A. W. Tsui · Y.-H. Chuang
Information and Communication Research Laboratories,
Industrial Technology Research Institute,
Rm. 513, Bldg. 51, 195, Sec. 4, Chung Hsing Rd.,
Chutung, Hsinchu 310, Taiwan

Y.-H. Chuang
e-mail: wood@itri.org.tw

H.-H. Chu (✉)
Graduate Institute of Networking and Multimedia,
Dept. of Computer Science and Information Engineering,
National Taiwan University,
No.1, Roosevelt Road, Sec.4,
Taipei 106, Taiwan
e-mail: hchu@csie.ntu.edu.tw

1 Introduction

The WiFi location system is a promising technology aimed at complying with real-world location-based applications. In comparison with other location systems using GPS, cameras, RFID, ultrasound, etc., WiFi location systems have several advantages in deployment practicability. For examples, WiFi works in both indoor and outdoor environments; it also leverages existing and widely-deployed Wi-Fi networks. As a result, several companies such as Ekahau (http://www.ekahau.com/), Aeroscout (http://www.aeroscout.com/), Innerwireless Pango (http://www.innerwireless.com/) and Skyhook wireless (http://www.skyhookwireless.com/) are actively developing location-based applications for using this technology in hospitals, warehouses, factories, amusement parks and other locations. Additionally, Cisco, a key WiFi industry player, has developed a unique WiFi localization system (http://www.cisco.com/go/location) based on WiFi access points that detect signal strengths emitted from client WiFi devices such that no software installation is needed on the WiFi devices.

Despite its numerous advantages in deployment practicability, deployment of current WiFi location systems remains problematic. A major issue is the WiFi hardware variance problem: the WiFi device used to train the radio map during the calibration phase (the training device) may differ from the WiFi devices used during the tracking phase (tracking devices). Varying Received Signal Strength (RSS) can degrade the signal patterns between training and tracking devices as well as the positional accuracy of WiFi location systems. The experiments described later in this paper indicate that the average positional error may increase by more than 100%. Further, this hardware variance problem is not limited to differences in the WiFi chipsets used by training and tracking devices (e.g., Intel Centrino vs. Lucent chipsets) but also occurs when the same WiFi chipsets are connected to different antenna types and/or packaged in different encapsulation materials (e.g., Intel Centrino chipset in a Sony VAIO laptop PC vs. the same chipset in a Panasonic laptop PC). Signal patterns are affected by both antennas and packaging materials. This hardware variance problem also applies to the Cisco location system: positional error is produced when the WiFi device used in the training phase emits signals of different signal strength from WiFi devices used in the tracking phase.

Of the several proposed solutions for addressing this hardware variance problem, the most effective is manual adjustment, proposed by Haeberlen et al. [1]. For RSS mapping between training and tracking devices, RSS readings are collected from both devices at the same location during the training phase. Experiments conducted by the authors and others [1, 2] show that RSS mapping from a tracking device to a training device exhibits a linear relationship. Hence, RSS mapping can be modeled as a linear function. The difficulty lies in manually identifying the best linear coefficients in a linear function that maps the RSS signal pattern of the target device to that of the training device. Although manual adjustment improves the positional accuracy of WiFi localization in hardware variance conditions, manually performing all possible combinations of pair-by-pair training for different WiFi training and tracking devices is overly laborious. Further, the ever-increasing number of new WiFi chipsets, antennas, and encapsulation materials makes this manual adjustment approach impractical for real-world deployment.

Ekahau [3] proposes an automated adjustment method in which signal transformation functions are learned by analyzing signal shifts from unknown tracking devices when they pass through certain easy-to-estimate locations, such as doorways, where signals exhibit highly distinguishable patterns. This method's primary drawback is that an unknown tracking device may not immediately enter these easy-to-estimate locations, or may stay only for a short time. Kjaergaard et al. [4] propose an improved automated adjustment method that uses the ratio of RSS values from different access points (rather than the absolute RSS values) in the location estimation to overcome the hardware variance problem. However, the method considers only the ratio term of the linear transformation function in the RSS mapping between two different WiFi devices, but not the offset term in the linear function. Hence, a sizable positional error occurs when RSS values of two WiFi devices vary mainly in the offset term.

We analyze this hardware variance problem and propose an unsupervised learning approach to automatically and accurately determine a linear transformation function that can map RSS signal patterns from any unknown tracking device to a training device, thus eliminating the need for exhaustive, manual pair-by-pair training. Further, in this work we demonstrate that unsupervised learning accurately and efficiently determines these transformation functions.

To put this work in the appropriate context, current fingerprint-based positioning systems have two phases: the off-line training phase during which RSS signals of a training device are used to calibrate a radio map, and the online estimation phase during which RSS fingerprints from a tracked device are used for positioning. As shown in Fig. 1, we propose the addition of an intermediate phase, called the online adjustment phase, during which the signal transformation function between the training device and a new tracked device is learned and determined. Adding this online adjustment phase reduces the positional error caused by the hardware variance between the training device and the tracked device. Note that the online adjustment phase is a one-time effort when a new tracked device, identified by its unique MAC address, first enters the environment. In subsequent visits of the same tracked device in the environment, the system uses its MAC address to find and reuse the previously trained transformation function.

Notable contributions of this study include the following: several unsupervised learning methods developed in this study resolve the WiFi variance problem by accurately and efficiently determining an RSS signal-pattern transformation function. These methods were implemented in a working WiFi positioning system, and performance was evaluated in an actual working environment. The performance of the proposed unsupervised learning method of RSS-based WiFi localization improved positional accuracy by as much as 46%. Additionally, when the tracking and training WiFi devices were identical, applying unsupervised learning to WiFi localization did not reduce positional accuracy.

The remainder of this paper is organized as follows. Section 2 presents the WiFi hardware variance problem by first demonstrating experimental results on the varying RSS signal patterns from different WiFi devices, and then the WiFi hardware variance problem is formulated. Section 3 presents the design and implementation of the proposed unsupervised learning method to address this WiFi hard-

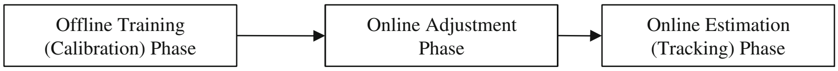


Figure 1 The online adjustment phase in in the WiFi fingerprinting-based localization system

ware variance problem while Section 4 describes the experimental setup and evaluation of the unsupervised learning methods in an actual working environment. In Section 5 we discuss related work, and in Section 6 we draw conclusions and suggest future studies.

2 Rationale

We first conducted experiments to identify varying RSS signal patterns between different WiFi devices. The experimental observations revealed how RSS signal-pattern variations cause positional error in a WiFi localization system. We formulate the proposed approach based on unsupervised learning to solve the RSS signal-pattern variation problem.

2.1 Signal-pattern variations of WiFi client devices

We performed experiments to determine the variation in RSS signal patterns from different WiFi devices. The test environment was the fifth floor of an office building. Figure 2 shows the floor plan of the test environment, which was 25 m by 47 m and had sixteen WiFi access points (APs). While walking the path indicated by the blue dotted line in Fig. 2, RSS readings for four different WiFi devices were recorded. Table 1 shows the hardware profiles of the four WiFi devices.

Table 1 WiFi client devices and hardware profiles

Abbreviations	WiFi client devices
IBM	IBM notebook PC with Intel PRO/wireless 2915 ABG
Compaq	Compaq WL110 wireless LAN PC card (Attacted to Fujitsu Tablet PC)
Orinoco	Orinoco wireless LAN PC card (attached to Fujitsu Tablet PC)
HP	HP iPAQ RW6828 PDA with built-in WiFi

Figure 3 compares the RSS readings of different training/tracking device pairs. Each point (RSS-x, RSS-y) on the plots represents RSS readings from two different devices at the same location and from the same WiFi AP. For example, if the Compaq device RSS readings are (x_{123}) from three WiFi APs (AP_{123}) while the HP device readings are (y_{123}) from the same three WiFi APs (AP_{123}) at the same location, the three points (x_{11}), (x_{22}), and (x_{33}) are plotted on the upper left graph of Fig. 3. Each of six plots in Fig. 3 is constructed from 500 sample points.

From the RSS trace data shown in Fig. 3, we can observe a linear shift in RSS readings between the two WiFi devices in all six tested pairs. For example, Fig. 3 shows the Compaq vs. HP RSS mapping relation, which can be approximated by a line with a slope of 0.93 at y-intercept (or “offset”) of 1.20. This linear shift in RSS signal patterns was also observed in experiments conducted by Kjaergaard et al. [4] and Haeberlen et al. [1].

Figure 4 plots the RSS readings of a Sony VAIO SZ18 laptop PC against a Panasonic CFT5 laptop PC at the same location. Both have the same Intel Centrino WiFi chipset but use different antennas and packing materials. The RSS analysis shows that their RSS signal patterns differ with an approximate linear RSS mapping function of a slope (0.92) at an offset (−8.155 dbm).

2.2 Effect of linear signal-pattern shift on the accuracy of a WiFi positioning system

Before describing how this linear shift in RSS signal patterns affects the positional accuracy of a WiFi position-

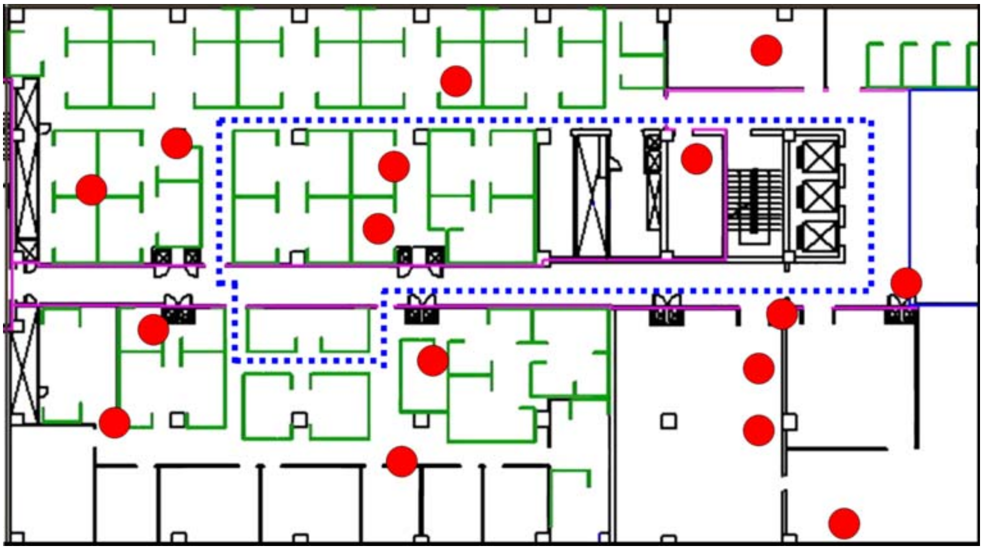


Figure 2 Sixteen WiFi APs (red circles), were distributed throughout the 25 m×47 m test environment. The movement of the tracking device is marked with a blue dotted line

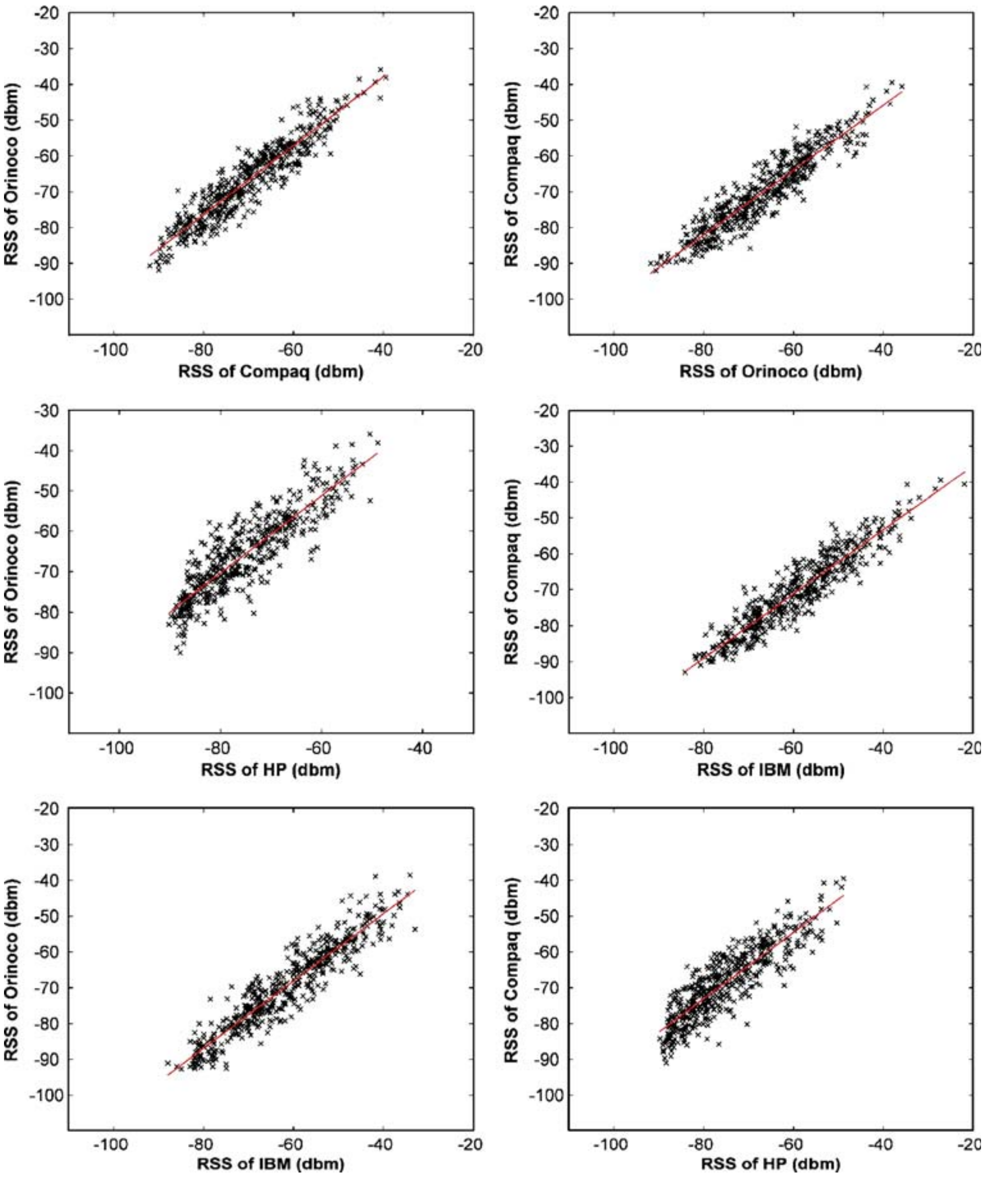


Figure 3 Correlation of several hardware pairs

ing engine, we examine the characteristics of a general RSS-based WiFi location system. Such a system consists of two phases. In Phase I, the offline training phase, we perform a site survey by using a training device to measure RSS signal patterns from different APs at fixed sample points in the environment. These RSS readings are encoded as fingerprints and recorded onto a radio map depicting the RSS of APs at different sample location points. Phase II is the online estimation phase, in which the location of tracking devices is calculated in real time by matching sample points on the radio map with the RSS fingerprint closest to the tracking device. By considering each RSS fingerprint a vector, the proximity of two RSS fingerprints can be measured using their Euclidean distance [5, 6] or a probabilistic model [1, 7–9].

The example in Fig. 5 shows the effect of RSS signal-pattern variations on the positional accuracy of RSS-based WiFi localization. The blue line (▴) indicates the RSS fingerprint of a training device measured at location x. The green line (▪) shows the RSS fingerprint of the same training device measured at location y. The red line (•) indicates the RSS fingerprint of the tracking device at location x. Since the tracking device differs from the training device in RSS signal patterns, the red (•) fingerprint exhibits a linear shift away from the blue (▴) fingerprint (Fig. 5). By computing their Euclidean distance, the red fingerprint vector is closer to the green fingerprint vector than to the blue fingerprint vector. Thus, the positioning system mistakenly estimates that the tracking device is located at y rather than at x.

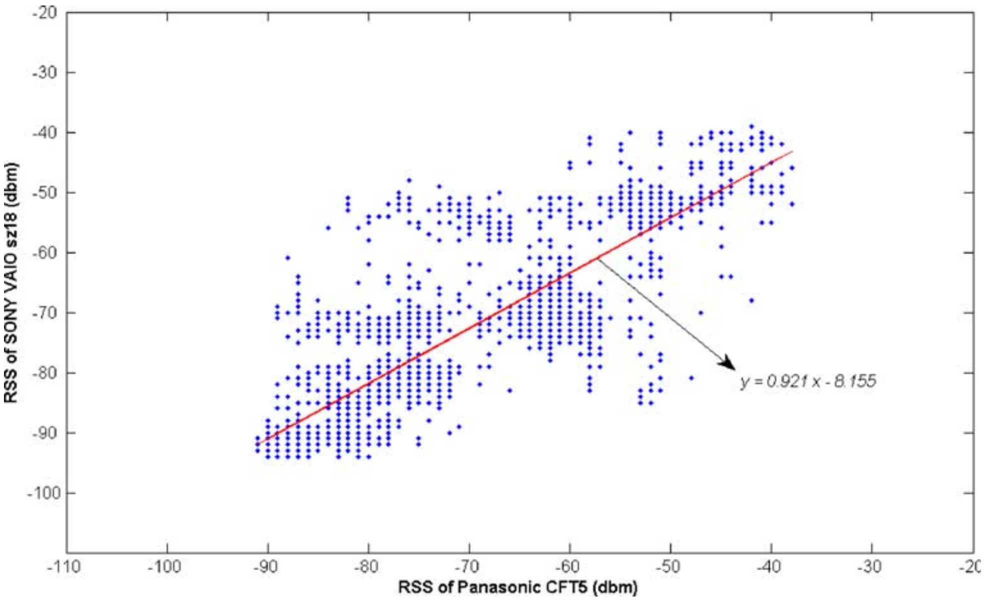


Figure 4 RSS signal patterns from Sony VAIO and Panasonic laptop PCs with the same Intel Centrino WiFi Chip

2.3 Signal-pattern transformation function

The proposed solution to the above problem is to apply a transformation function on the RSS fingerprint of the tracking device such that the transformed fingerprint (the grey dotted line in Fig. 5) is shifted closer to the RSS fingerprint of the training device. Thus, the positioning engine can estimate the tracking location of the tracking device.

This hardware variation problem is formally defined as follows. Denote the training device as H_c (c for calibration) and the tracking device as H_o (o for observed). Since H_c and H_o produce different RSS signal patterns, denoted as S_c and S_o, the probability of training RSS readings at location l using training device H_c, denoted as P (S_c | Location = l, Hardware = H_c), differs from the one obtained for the tracking device, denoted as P (S_o | Location=l, Hardware=H_o) . The problem is finding an accurate transformation function F such that applying F to S_o shifts P (S_o | Location=l, Hardware=H_o) closer to P (S_c | Location=l, Hardware=H_c) for all li in the tracking space,

$$\underline{S_c} = F\left(\underline{S_o}\right) \tag{1}$$

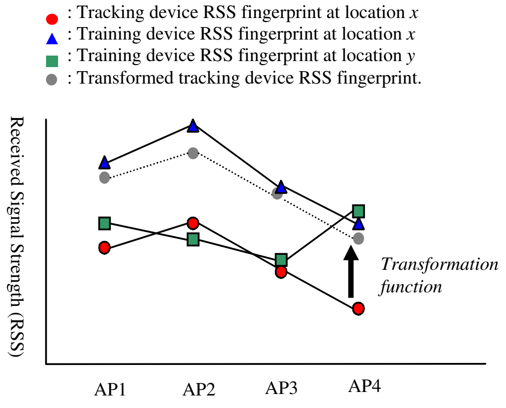

● : Tracking device RSS fingerprint at location *x*
▲ : Training device RSS fingerprint at location *x*
■ : Training device RSS fingerprint at location *y*
● : Transformed tracking device RSS fingerprint.
Received Signal Strength (RSS)
Transformation function
AP1
AP2
AP3
AP4

Figure 5 Example of positional estimation error caused by RSS signal-pattern variation in RSS-based WiFi localization

A simple method of determining this transformation function is to survey the site by collecting RSS samples from devices H_c and H_o while manually holding them at each location. Using this training dataset as examples of RSS mappings between tracking and training devices, different learning algorithms are applied to learn a signal-pattern transformation function. From a learning perspective, this manual procedure of collecting training examples is analogous to manual labeling inputs in supervised learning. However, as mentioned in Section 1, this manual data collection is impractical given a large number of WiFi device pairs. Therefore, an unsupervised learning algorithm is needed.

2.4 Unsupervised learning

The proposed unsupervised learning system automatically learns this signal-pattern transformation function at runtime for any unknown tracking device. The learning procedure consists of the following two general steps.

- The RSS readings from an unknown tracking device are first labeled with a rough location estimation using a correlation ratio computed from the Pearson product-moment correlation coefficient [10] defined below:

$$r = \frac{\sum_{i=1}^{k}\left(\left(s_c^i - \mu\left(\underline{S_c}\right)\right)\left(s_o^i - \mu\left(\underline{S_o}\right)\right)\right)}{\sqrt{\sum_{i-1}^{k}\left(s_c^i - \mu\left(\underline{S_c}\right)\right)^2}\sqrt{\sum_{i=1}^{k}\left(s_o^i - \mu\left(\underline{S_o}\right)\right)^2}} \tag{2}$$

The k is the number of APs (or the dimension of RSS fingerprint vectors), s_c is the RSS fingerprints of the training device from the radio map, μ is the mean or expected value of a random variable, and s_o is the RSS fingerprint from the tracking device. The absolute value of Pearson correlation ratio has a value range in (0–1) where 1 indicates the best linear dependency (or greatest similarity between two fingerprint vectors), and 0 indicates complete linear independency (or least similarity). The Pearson correlation ratio is therefore used to measure similarities in RSS fingerprints between tracking and training devices.

- After labeling the RSS data with rough location estimates, four learning algorithms including linear regression, two versions of expectation maximization, and neural networks, are applied to train the transformation function. These four learning algorithms are detailed in Section 3.

The proposed approach differs from the one proposed by Kjaergaard et al. [4], which computes an RSS ratio (i.e., the RSS reading of one AP divided by the RSS reading from another AP) to reduce the effect of linear shift in RSS fingerprint matching. Since the Kjaergaard approach only approximates the ratio term but not the offset term in a linear shift function, approximation error is increased when the offset term is relatively large. In comparison, the proposed approach uses the Pearson formula in Eq. (2), which captures both the ratio and the offset terms in a linear relationship.

3 Design and implementation

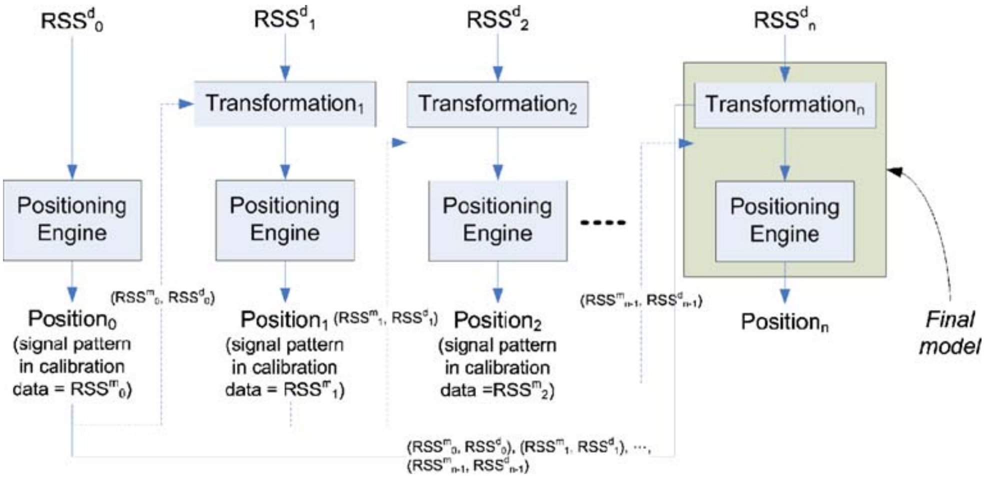
Our system design is shown in Fig. 6. In the first step, RSS fingerprints (RSS^d) measured from an unknown tracking device are sent to the localization system server. In the second step, Eq. (2) is applied to compute the Pearson correlation ratio from the RSS fingerprints. This ratio is then used as input to the positioning engine, which consults the radio map to find the best matched location point. Since the parameters to the transformation function have yet to be determined at this time, location estimate is rough. In the third step, from the rough location estimate, the corresponding RSS fingerprints stored on the radio map (RSS^m) are identified and coupled with the device's RSS fingerprints (RSS^d) to form a training sample (RSS^d, RSS^m). Assuming that n training samples are collected over the training period, the training set consists of $\left\{\left(RSS_0^d, RSS_0^m\right),\left(RSS_1^d, RSS_1^m\right),\left(RSS_{n-1}^d, RSS_{n-1}^m\right)\right\}$. We propose two learning methods to train this transformation function – batch learning and iterative learning. In batch learning (represented by the solid path in Fig. 6), the entire training set is input to a learning algorithm at the end of the training period to determine the parameters for the transformation function. In iterative learning (represented by the dotted path of Fig. 6), individual training data is input iteratively to a learning algorithm to determine the parameters for the transformation function. Although the initial parameters to the transformation function are likely to be inaccurate due to the training set's small size, they will gradually become more accurate through repetitive training. During the training process, users may move around freely: there is no need to stand still at certain locations as in the calibration process.

3.1 On-line regression algorithm

The linear regression in the first learning algorithm assumes that the regression model for a tracking device's RSS mappings to a training device exhibits a linear relationship, as shown in Fig. 3:

$$RSS_c = b + a \times (RSS_o) + \varepsilon. \tag{3}$$

RSS_c and RSS_o represent RSS fingerprints from the training and tracking devices, *(a, b)* are coefficients in the


$RSS^d{}_0$
$RSS^d{}_1$
$RSS^d{}_2$
$RSS^d{}_n$
Transformation$_1$
Transformation$_2$
Transformation$_n$
Positioning Engine
Positioning Engine
Positioning Engine
Positioning Engine

($RSS^m{}_0$, $RSS^d{}_0$)
($RSS^m{}_{n-1}$, $RSS^d{}_{n-1}$)
Position$_0$ (signal pattern in calibration data = $RSS^m{}_0$)
Position$_1$
($RSS^m{}_1$, $RSS^d{}_1$)
(signal pattern in calibration data = $RSS^m{}_1$)
Position$_2$ (signal pattern in calibration data =$RSS^m{}_2$)
Position$_n$
Final model
($RSS^m{}_0$, $RSS^d{}_0$), ($RSS^m{}_1$, $RSS^d{}_1$), ···, ($RSS^m{}_{n-1}$, $RSS^d{}_{n-1}$)

Figure 6 System design

transformation function, and ε is the error term. The training dataset is determined by the tracking device's RSS readings and are labeled with rough position estimates using the Pearson correlation ratio. After collecting a sufficient number of data points, least squares analysis is applied to find the best *(a, b)* coefficients with the lowest sum of error squares.

3.2 EM algorithm

An alternative approach to on-line regression is expectation maximization (EM). Starting with the initial estimates of *(a, b)* coefficients, EM is an iterative procedure to refine the *(a, b)* estimates by repeatedly computing expectation and maximization.

The true location is postulated by combining the original probabilistic location computation with the Pearson correlation ratio. Particle filters [11–14] are used for historical moving pattern adjustments with a probability of correlation coefficient for each cell. At each iteration, the estimated linear parameters *(a, b)* are used in the subsequent iteration to transform the RSS fingerprints from the tracking device. This iterative procedure continues until a convergence criterion is met.

The convergence criterion is that the result of the linear mapping function (i.e., the 'y' in '$y=ax+b$'), stays within the range of δ (=1) dbm over a period of time t (=50 s). We choose the result of the linear mapping function (y) over the parameters (a, b) for faster convergence. Since the RSS values have limited range between −30 dbm and −90 dbm, training points fall only within a small segment of this linear function. Therefore, it is more important that we accurately identify this specific segment rather than the entire linear function. A more detailed discussion on the convergence criterion is discussed in Section 4.3.

The output parameters at each iterative step are the linear coefficients *(a, b)*. The Θ = (a, b) and $\underline{S_o}(t)$ are denoted as the tracking device's RSS readings at time t. The optimization problem is as follows:

$$\underset{\Theta}{argmax}\, P\left(\Theta|\underline{S_o}(t)\right) = \underset{\Theta}{argmax}\left\{\frac{P\left(\Theta, \underline{S_o}(t)\right)}{P\left(\underline{S_o}(t)\right)}\right\}. \qquad (4)$$

EM is applied to solve the above problem by estimating $P\left(l_i|\underline{S_o}(t), \Theta(t)\right)$ and maximizing Θ(t+1), where $l_i \in L$ (L for the set of all locations in space) is the latent variable.

3.3 Neural network algorithm

We implemented the neural network learning method to capture any non-linear components of the signal-pattern transformation function. Similar to online regression, the training dataset for the neural network was obtained from the tracking device's RSS readings and labeled with rough position estimates from the Pearson correlation ratio. The network realized using radial basis functions [15] has the following form:

$$\varphi(\underline{S}) = \sum_{i=1}^{N} w_i \rho(\|\underline{S} - c_i\|), \qquad (5)$$

where N is the number of neurons and S is the single signal-strength pattern value observed. The basic function $\rho(\|\underline{S} - c_i\|)$ is Gaussian:

$$\rho(\|\underline{S} - c_i\|) \propto \exp[-\beta\|\underline{S} - c_i\|]. \qquad (6)$$

The weights w_i are learned using gradient descent:

$$w_i(t+1) = w_i(t) + v[y(t) - \varphi(\underline{S}(t))]\rho(\|\underline{S}(t) - c_i\|), \qquad (7)$$

where *y(t)* is the signal pattern with the highest correlation coefficient selected from the training data.

3.4 Extended EM algorithm

Since both EM and neural networks are limited by local optimum, we designed and implemented a fourth method to extend EM as follows. First, six EM models with different initial seeds are executed in parallel. From the six EM models, the results with the highest probability are used as the output [16].

4 Evaluation

This section describes the experimental procedure and analyzes the performance results of the unsupervised learning system in an actual WiFi localization system and environment.

4.1 Experimental setup

As Fig. 2 shows, the test environment was the same as in the previous experiment for observing signal-pattern variations from different WiFi devices. The radio map was constructed as follows. First, the test environment was divided into 38*69 cells with each cell occupying a 0.69-meter by 0.69-meter space. The position of the tracking devices was estimated using cell granularity. Additionally, to minimize manual calibration, 107 of the 2,622 cells were uniformly selected as the training cells. Thirty RSS samples were collected from each training cell. The RSS samples from the other unselected cells were interpolated with those of the training cells. Interpolation was intended to reflect the common practice of minimizing manual calibration in deployments over a wide coverage area, even though interpolation reduces positional accuracy [17].

4.2 Positional accuracy

Table 1 shows the four WiFi devices used in the experiments. The Compaq device was selected as the training device for calibrating the radio map. During the online estimation phase, a person carried each of these four WiFi devices while walking at a constant speed along a path indicated by the blue dotted line in Fig. 2. The RSS readings along this walking path were collected for use in the training dataset for each of the four learning algorithms (that is, online regression, EM, neural network, and extended EM) to train the signal-pattern transformation function.

Figure 7 plots the mean positional error when applying each of four learning algorithms to track each of the four different WiFi devices, and also plots the mean positional error for manual adjustment and non-adjustment. Manual adjustment re-implements the method developed by Misikangas et al. [3], which provides near-optimal performance. Non-adjustment means no transformation function was applied for WiFi localization; therefore, its performance is used as a baseline for measuring improvements in the four learning algorithms.

From Fig. 7 we find that: (1) the EM produced the least average positional error of the four tracking devices, and its performance approached that of the near-optimal manual adjustment; (2) except for the Orinoco device, which had an RSS signal pattern similar to that of the Compaq training device, the learning algorithms all yielded less positional error than non-adjustment. This suggests that applying the unsupervised learning to RSS-based WiFi localization effectively reduced positional error due to hardware variance. (3) Under some training/tracking device pairs, EM outperformed the near-optimal manual adjustment. One explanation is that the unsupervised system adjusts dynamic changes in environmental factors (for example, humidity levels, open/closed doors, etc.) that affect signal patterns.

4.3 Training time

In EM and neural networks, the training time is important for determining a signal-pattern transformation function for an unknown tracking device. During the learning phase, positional estimates are unreliable.

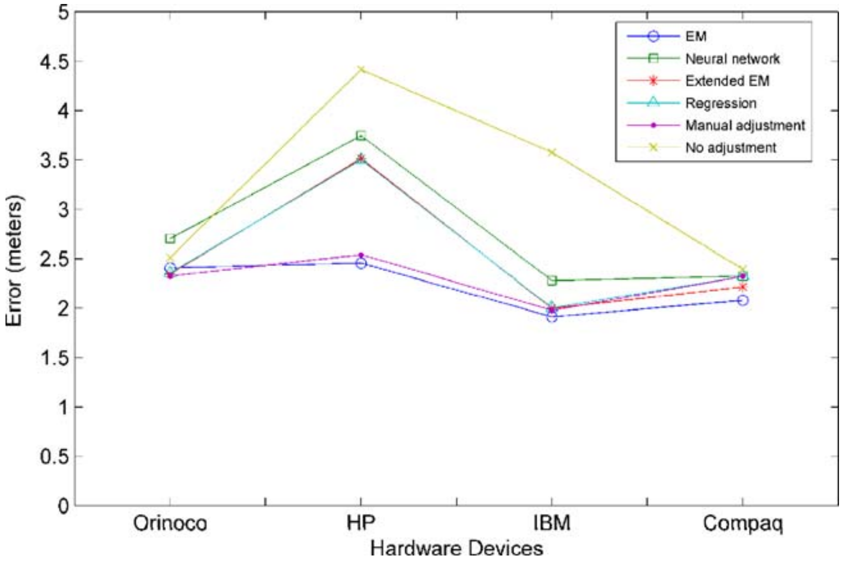
The training process is complete when the changes in the linear parameters of the transformation function being trained fall within a limited range. However, the experiments reveal that while function parameters rarely converge, the output of transformation functions often do. The reason is that several solutions to function parameters may co-exist when the input training dataset is concentrated on a small segment of the function where the RSS readings fall between −90 dbm and −30 dbm. To address this problem, the convergence criterion depends on the change in function output rather than the change in the trained input parameters.

The convergence criterion uses the near-optimal output from manual adjustment (as determined during post-processing) as a baseline to determine the speed of the proposed runtime learning algorithms (EM or neural network) which then stabilized at the near-optimal output. Specifically, as Fig. 8 shows, the training curves were constructed using the near-optimal output differences and the learning algorithm.

$$\left|output_{\ manual\ adjustment} - output_{\ learning\_algorithm}\right|, \tag{8}$$

where the tested learning algorithms were EM (dotted line) and neural network (solid line), and the training/tracking

Figure 7 Positional error (in meters) after applying four different learning algorithms (online regression, EM, neural network, and modified EM) to four different tracking devices (Orinoco, HP, IBM, and Compaq). The Compaq was the training device



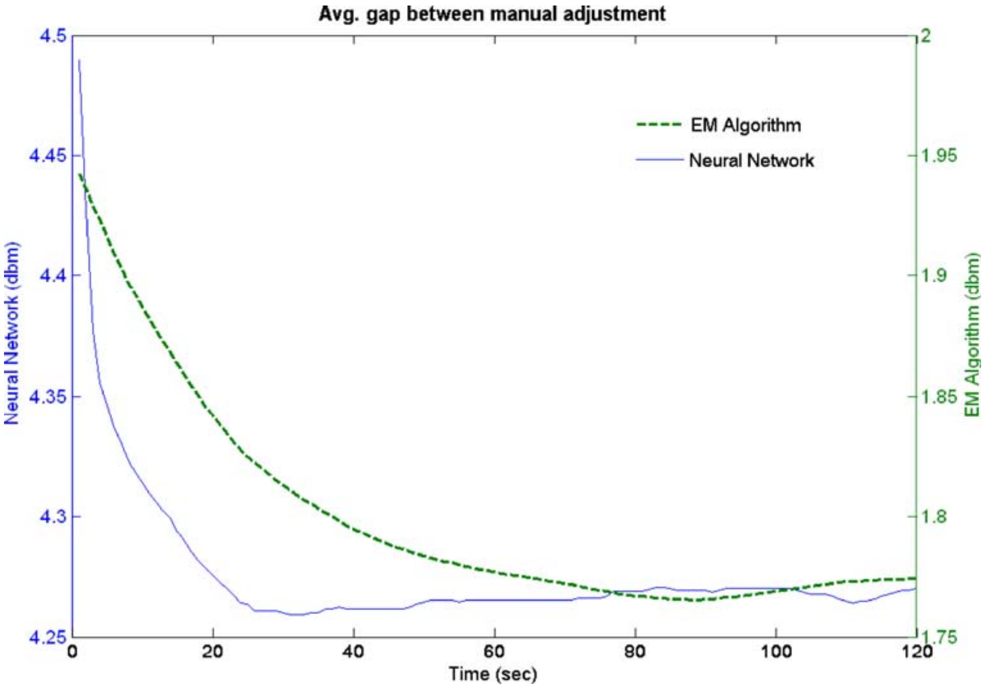


Figure 8 Comparison between convergence times

devices were Compaq/HP. The training times of both the EM and neural network were under 100 s. Although the EM training time was slightly longer than that of the neural network, the EM training error was smaller (1.8 dbm vs. 4.2 dbm).

Whether or not the 100-second training time is sufficient depends on the specific application. For example, 100 s may be acceptable for a location-based museum tour guide but may be too long for locating an emergency call. Note that the 100-second training time is a one-time cost when a new WiFi device enters an environment. After the system learns the parameters to its transformation function, it stores them into a database indexed by the device's MAC address. Subsequent visits from the same WiFi device incur a database lookup and require no additional learning cost.

4.4 Case study: the Orinoco as the training device

After using the Compaq computer as the training device in the above tests, the unsupervised learning system was tested using other training devices. Table 2 shows the amount and percentage of positional error reduction when the Orinoco was used as the training device.

Table 2 Positional error reduction when using orinico as the training device

Tracked device	Positional error reduction (meters)	Positional error reduction (%)
Compaq	0.31	13.13
IBM	1.66	46.51
HP	1.96	44.41

The experiments show that the IBM device produced the strongest RSS readings followed by the Orinoco and Compaq devices (which yielded RSS readings with similar strengths) and finally the HP device, which recorded the weakest strength. Since the difference in RSS signal patterns between Compaq (the tracking device) and Orinoco (the training device) was small, the 0.31 m positional error reduction was also small. Conversely, the difference in RSS signal patterns was large for Orinoco-IBM and Orinoco-HP pairs, as was the positional error reduction (1.66 m and 1.96 m, respectively). Table 2 shows that, given a larger RSS signal-pattern difference between tracking/training devices, the proposed system generally achieves higher positional error reduction. However, one exception is discussed in Section 4.7 below.

4.5 Case study: similarity between tracking/training devices

In this case study we investigated whether unsupervised learning on the same training/tracking device pairs degrades positional accuracy given that learning and applying transformation functions to the same device pairs is unnecessary. Table 3 shows the positional error for Orinoco and Compaq training and tracking devices, respectively, with and without using unsupervised learning. No degradation of positional accuracy was observed. Surprisingly, a slight improvement was observed (i.e., the positional error was reduced from 2.66–2.40 m for the Orinoco devices and from 2.40–2.08 m for the Compaq devices). A possible

Table 3 Positional error reduction in the same training/tracking device pair with/without learning

	Positional error without learning (meters)	Positional error with EM learning (meters)
Orinoco	2.66	2.40
Compaq	2.40	2.08

explanation is that, since training was performed 1 day before testing, the unsupervised system also adjusted to changes in environmental factors (for example, humidity levels, open/closed doors, the presence of people, etc.) that affect WiFi signal patterns. This effect was also observed in previous work [18–21] that applied learning algorithms to adjust the radio maps given sensor-detected changes in these environmental factors. However, in the future we are interested in validating how well unsupervised learning can adapt to such changes in the environmental factors.

4.6 Case study: variable-speed vs. constant-speed movement

Since variable-speed movement is common in real-world scenarios, in this case study we compared the effectiveness of unsupervised learning for both constant-speed and variable-speed movements. The variable-speed trace was collected as follows. Figure 9 shows several walking segments marked with arrowed lines. Each walking segment has a unique movement speed whereas the speed within each walking segment is constant.

Since most WiFi localization systems adopt a motion model that assumes constant-speed movement, variable-speed movement often increases positional error. Table 4 shows positional error after applying different learning methods to WiFi localization in two cases involving constant-speed movement and variable-speed movement. The results in Table 4 show that the unsupervised learning method is effective for both constant-speed and variable-speed movement.

4.7 Relationship between RSS signal-pattern difference and positional error

Intuitively, since increased difference in RSS signal patterns between tracking and training devices produces larger difference between the tracking device's RSS fingerprint and the RSS fingerprint on the radio map constructed by the training devices, the positional error in WiFi localization should be larger also. Although this relationship between the RSS signal-pattern difference and positional error seems reasonable, the experiments in this study proved otherwise.

Figure 10 plots this relationship between the positional error and RSS signal-pattern variance for several training/tracking device pairs. The *x*-axis measures the percentage of the average AP RSS readings between the training and tracking devices, computed as follows:

$$\frac{|avg(RSS_c - avg(RSS_o)|}{avg(RSS_o)}. \tag{9}$$

The *y*-axis measures the average percentage of increased positional error due to device hardware variance, which is computed as the difference between the positional error without hardware variance (that is, using the same device for tracking and training) and the positional error with hardware variance. The results in Fig. 10 do not indicate

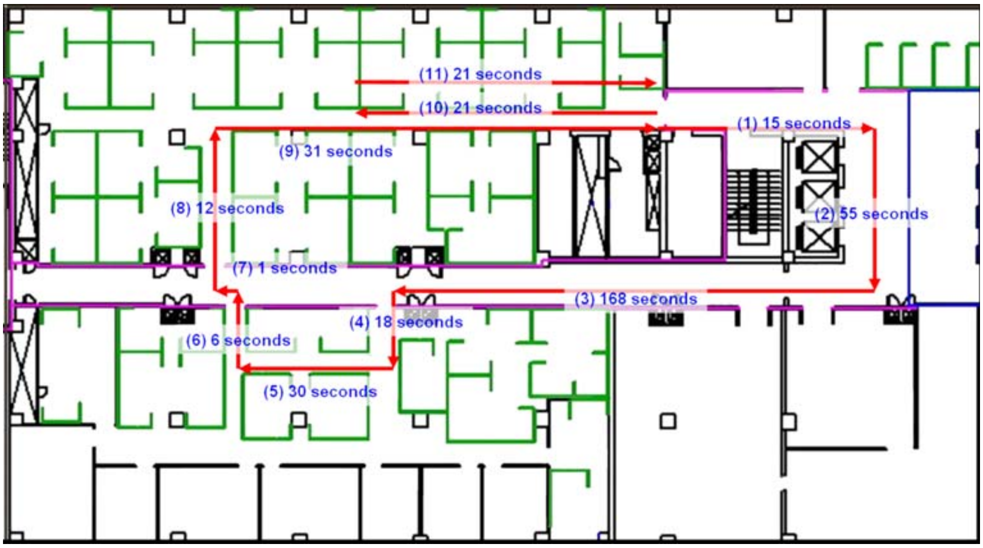


Figure 9 Moving testing traces

Table 4 Positional error under constant-speed movement and variable-speed movement

	EM	Neural network	Modified EM	Online regression
Constant-speed motions	2.46	3.75	3.51	3.50
Variable-speed motions	2.37	3.82	2.48	3.11

that greater RSS signal-pattern variance increases positioning error.

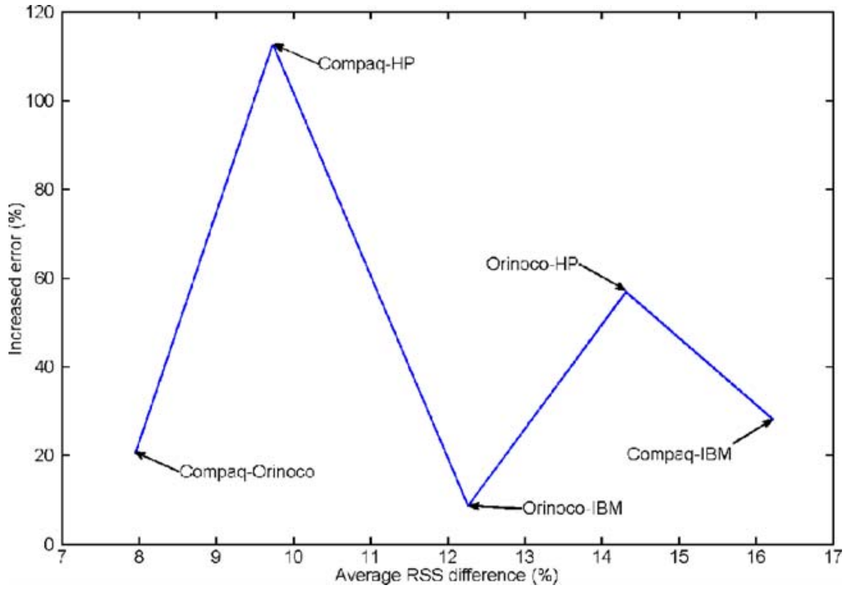
Several factors such as AP spatial distribution can enlarge or mitigate the effect of RSS signal-pattern variance on positional error. An interesting situation in which a large RSS signal-pattern variance produces negligible positional error is the following. This situation arises under two conditions: (1) the RSS readings between the tracking and training devices differ only in offset but not in ratio, and (2) the spatial distribution of APs is balanced or evenly distributed relative to the position of the tracking device. Figure 11 shows an example of a balanced distribution of APs, which are marked in red circles, relative to the location of a tracking device, which is moving within the blue dotted rectangular area. The distribution is balanced to the tracking device in that the APs to its right and left are more or less equal in number and distance. Appendix describes an analytical model showing how this balanced AP distribution mitigates the effect of RSS signal-pattern variation on positional accuracy.

5 Related work

Many location estimation systems have been developed using Wi-Fi RSS values to estimate locations. These systems can be categorized into two broad approaches. The first approach is based on the deterministic method [5, 6]. Systems following this approach apply deterministic inference, such as triangulation and *k*-nearest-neighbors (KNN) search, to estimate the target device's location. For example, the RADAR system applies KNN to obtain the *k* nearest neighbors and estimates the location of the target device by averaging the locations of these *k* nearest neighbors. The second approach is based on the probabilistic method [7–9, 11]. Seshadri et al. [11] apply Bayesian inference, which uses multiple probabilistic models and histograms to enhance the performance of the original system. It calculates the conditional probabilities over location-based RSSI. It also adds a motion model to describe the continuity in human movement such that it can lower the oscillatory location estimations in Wi-Fi based localization systems.

A notable WiFi based localization system is from Cisco: this system tracks all WiFi devices in the environment without installing any additional software on the devices. WiFi access points are programmed to measure RSS signals emitted by the tracked devices. Then, RSS measurements from multiple access points are aggregated at a localization server that determines the locations of all tracked devices. Similar to other WiFi fingerprint-based localization systems, the Cisco system requires an offline training phase during which a tracked device is used to calibrate a radio map. Therefore, our unsupervised learning approach is also

Figure 10 RSSI difference vs. decreased positional accuracy



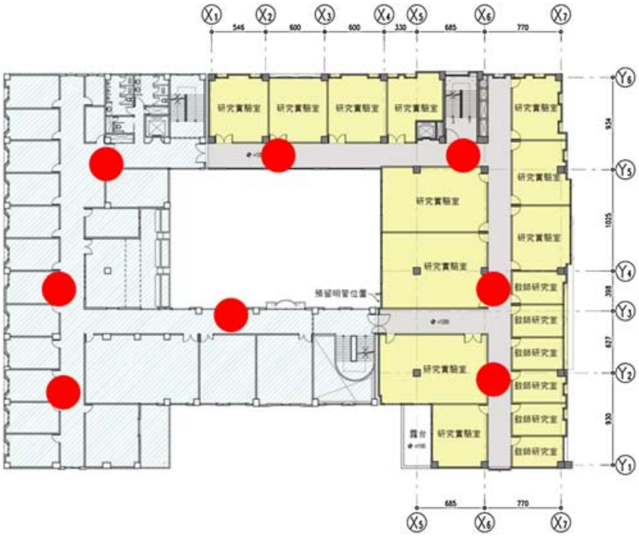


Figure 11 A balanced AP distribution that mitigates the effect of RSS signal-pattern variation on positional accuracy. The APs are marked in red circles. The tracking device is moving within the blue rectangular area

applicable to address the hardware variance problem in the Cisco system. Additionally, since our unsupervised learning algorithms are computed on the server side, it works well with the Cisco system in that no software installation is needed on the WiFi devices.

Hardware variance can be minimized by not relying on RSS information to implement WiFi positioning systems. For example, Cheng et al. [22] develop a method of ranking (i.e., high to low) RSS values instead of relying on absolute RSS values. Ranking overcomes the hardware variance problem because it is not effected by a linear shift in RSS. However, the ranking approach sacrifices positional accuracy because the detailed data for signal strength levels is not used.

This hardware variance problem was addressed by Misikangas et al. [3], who propose a manual approach based on pair-wise mapping. In this approach, different hardware devices are placed at the same position at the same time to differentiate their signal patterns; their mapping functions are then derived. Haeberlen et al. [1] propose a similar manual solution. Their experiments demonstrated a linear RSS mapping function between different WiFi devices. They then used the least-squares fit method to determine the coefficients in the linear transformation function. However, they did not apply the signal-pattern transformation function in a positioning engine to demonstrate improved positional accuracy. Although manual adjustment can achieve good positional accuracy, its main problem is the required manual labor, which does not scale well with the size of the environment and the wide array of emerging WiFi devices.

Misikangas et al. [3] describe an automatic approach for solving this hardware variance problem. Firstly, they manually obtain the available hardware pairs. They then assume the existence of an easily distinguishable location (i.e., a location with a unique RSS fingerprint) such as the entrance to a room. When an unknown WiFi device moves through this location, its signal patterns are collected, and a small set of calibrated-observed variant signal pairs are produced. Instead of computing the transformation from this limited information, the best-fit transformation identified in the manually compiled transformation database is assigned to this unknown device. However, an easily distinguishable location is required, which may only exist in certain environments. Additionally, a new WiFi device may never pass through these easily distinguishable locations or may not pass through them early enough to learn its transformation function. Haeberlen et al. [1] also explore fully automatic calibration. They suggest EM and particle filtering to learn coefficients, but do not mention implementation details or experiments. Kjærgaard et al. [4] propose an automatic method using RSS ratios when matching closeness in the RSS fingerprints. Although using the RSS ratios reduces the linear shift effect in RSS fingerprints caused by hardware variance, it can still produce errors (as described in Section 2) when the offset component in the linear shift is large.

Some WiFi location systems [23] do not use RSS fingerprints for location determination; for these systems, therefore, hardware variance does not present a problem. One such system, the Aeroscout WiFi location system, is based on accurately computing the TDOA (time difference of arrival) of WiFi signals. In order to accurately measure the time differences, specialized and proprietary hardware is needed to tightly synchronize the clocks on the WiFi devices. NearMe [23] also avoids the hardware variance problem by using only the received access points for proximity determination. Since its goal is to compute not fine-grained location but proximity among devices, the detailed RSS information is not needed.

6 Conclusion and future work

This work presents an unsupervised learning method for solving the hardware variance problem in WiFi localization. At runtime, the unsupervised learning method automatically learns a transformation function for mapping WiFi signal patterns from an unknown tracking device to the training device under which the radio map was calibrated. Several learning algorithms, including online regression, EM, neural network and extended EM were designed, implemented, and evaluated in a working WiFi localization system and environment. The experimental results demonstrate that, in WiFi localization, applying a transformation function learned from our unsupervised learning reduces

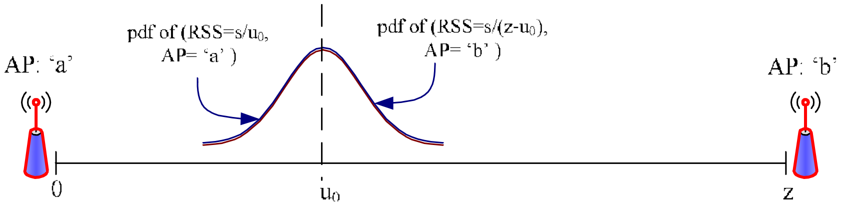


Figure 12 Highest probability distribution at location u_0

position error caused by device hardware variance by an average of 4%. Additional case study experiments show that (1) positional accuracy in the same training/tracking device pairs does not degrade, and (2) unsupervised learning is effective for both variable-speed and constant-speed movement. Finally, RSS signal-pattern variance was shown to be uncorrelated with positioning error.

Several questions remain for future studies in unsupervised learning. First, this technique could be applicable to other RSS-based localization systems such as Zigbee, WiMax, and GSM, in reducing positional error caused by hardware variance. Second, this unsupervised learning technique could also be applicable in adapting WiFi localization to dynamic environment factors affecting WiFi signal patterns such as humidity level, human presence, open/closed doors, etc.

APPENDIX: Analytical model for the balanced AP distribution

A typical location in an RSS fingerprint localization system is characterized by several (RSS, AP) pairs, where RSS is not a single value but rather a distribution of signals collected from the training phase and often modeled by a Gaussian distribution. While tracking a device, the probability of a set of observed (RSS, AP) pairs against a certain location is then computed by multiplying all the probabilities acquired from the previously modeled Gaussian probability distribution function. The location with the highest joint probability is the output result.

The above localization system is assumed here. Further, without loss of generality, the following assumptions are made:

1. RSS decays linearly;
2. The variances of all pairs are identical.

Consider the one-dimensional example in Fig. 12. Two access points AP_a and AP_b are at either side of a tracking device. Suppose the tracking device is at an arbitrary position u_0 on the line from 0 (the leftmost position) to z (the rightmost position).

According to the first assumption above, if the RSS directly beneath an access point is s, the distributions of $(RSS{=}s/u_0,\ AP{=}AP_a)$ and $(RSS{=}s/(z\text{-}u_0),\ AP{=}AP_b)$ at position u_0 are identical, and u_a and u_b are located at u_0. If the above two RSS signal patterns are entered into an RSS-based positioning engine, the estimated location will be u_0.

As Fig. 13 shows, if the tracking device differs from the training device with a linear RSS mapping function with slope = 1 as in the first assumption, the RSS distribution is simply shifted.

Although u_0 is no longer the most probable location for both APs, the multiplied probability is still the highest. This outcome is demonstrated by comparing the multiplied probability at each position. Since the RSS variances are assumed identical, in the p.d.f. of Gaussian distribution $\frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$, we need only compare the $(x - u)^2$ part. Restated, the smaller the value, the higher the probability.

At u_0, after multiplication, the next procedure would be

$$[u_0-(u_0-b)]^2+[u_0-(u_0+b)]^2=2b^2.$$

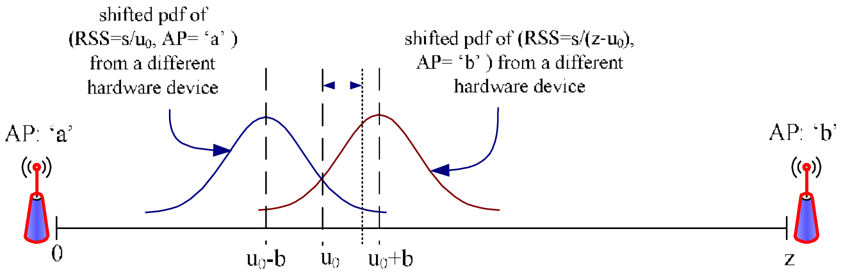

shifted pdf of (RSS=s/u0, AP= 'a') from a different hardware device
shifted pdf of (RSS=s/(z-u0), AP= 'b') from a different hardware device
AP: 'a'
AP: 'b'
0
u0-b
u0
u0+b
z

Figure 13 Highest probability distribution at location u_0 shifted by hardware difference with linearity of $y=x+b$

Assume an arbitrary position denoted as (u_0+d) in Fig. 13. After substitution, $(x - u)^2$ becomes

$$[(u_0+d)-(u_0-b)]^2+[(u_0+d)-(u_0+b)]^2$$
$$=2\left(d^2+b^2\right).$$

Since $2(d_2 + b_2) > 2b_2$ for any nonzero d, u_0 is the location with maximum probability.

References

1. Haeberlen A, Flannery E, Ladd AM, Rudys A, Wallach DS, Kavraki LE (2004) Practical robust localization over large-scale 802.11 wireless networks, in Proceedings of ACM MOBICOM, pp. 70–84
2. Tao P, Rudys A, Ladd AM, Wallach DS (1996) Wireless LAN location–sensing for security applications, In Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence, pp. 11–20
3. Misikangas P, Lekman L (2003) Applications of signal quality observations, Patent Cooperation Treaty (PCT) WO 2004/008796 A1
4. Kjaergaard M, Munk C (2008) Hyperbolic location fingerprinting: a calibration-free solution for handling differences in signal strength, In Proceedings of IEEE International Conference on Pervasive Computing, pp. 110–116
5. Bahl P, Padmanabhan VN (2000) RADAR: An in-building RF-based user location and tracking system, In Proceedings of IEEE Infocom, pp. 775–784
6. Bahl P, Padmanabhan VN, Balachandran A (2000) Enhancements to the RADAR User Location and Tracking System, Microsoft Research Technical Report: MSR-TR-00-12, February 2000
7. Ladd AM, Bekris KE, Rudys A, Marceau G, Kavraki LE, Wallach DS (2002) Robotics-based location sensing using wireless Ethernet, in Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking (MOBICOM), pp. 227–238
8. Roos T, Myllymaki P, Tirri H, Misikangas P, Sievanan J (2002) A probabilistic approach to WLAN user location estimation. Int. j. wirel. inf. netw. 9(3):155–164, July 2002. doi:10.1023/A:1016003126882
9. Thrun S (2000) Probabilistic algorithms in robotics. AI Magazine 21(4):93–109
10. Pearson K (1900) Mathematical contribution to the theory of evolution VII: on the correlation of characters not quantitatively measurable. Philos. Trans. R. Soc. Lond. Ser. A: Math. Phys. Sci. 195:1–47 doi:10.1098/rsta.1900.0022
11. Seshadri V, Zaruba GV, Huber M (2005) A Bayesian sampling approach to in-door localization of wireless devices using received signal strength indication, in Proceedings of IEEE International Conference on Pervasive Computing (PERCOM 2005), pp. 75–84
12. Hightower J, Borriello G (2004) “Particle filters for location estimation in ubiquitous computing: a case study,” In Proceedings of Ubiquitous Compuating, pp. 88–106
13. Schulz D, Fox D, Hightower J (2003) People tracking with anonymous and ID-sensors using Rao-Blackwellised particle filters, in Proceedings of IJCAI, pp. 921–926
14. Fox D, Hightower J, Liao L, Schulz D, Borriello G (2003) Bayesian filtering for location estimation, in Proceedings of IEEE Pervasive Computing, pp. 24–33
15. Michell T (1993) Machine Learning, McGraw Hill, pp. 239–240
16. El-Gallad A, El-Hawary M, Sallam A (2001) Swarming of intelligent particles for solving nonlinear constrained optimization problem. Int J Eng Intell Syst 9(3):155–163 Sept. 2001
17. Krishnan P, Krishnakumar A, Ju W-H, Mallows C, Ganu S (2004) A system for LEASE: Location estimation assisted by stationary emitters for indoor RF wireless networks, in Proceedings of IEEE INFOCOM, pp. 1001–1011
18. Chen Y-C, Chiang J-R, Chu H-H, Huang P, Tsui AW (2005) Sensor-assisted Wi-Fi indoor location system for adapting to environmental dynamics, in Proceedings of ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM 2005), pp. 118–125
19. Lin T-H, Chen J-P, Chen H-H, Huang P, Chu H-H (2006) Enabling energy-efficient and quality localization services, in Proceedings of the IEEE International Conference on Pervasive Computer and Communications (PERCOM 2006), Work In Progress Session, pp. 624–627
20. Chan L-W, Chiang J-R, Chen Y-C, Ke C-N, Hsu J, Chu H-H (2006) Collaborative localization—enhancing WiFi-based position estimation with neighborhood links in clusters, in Proceedings of the International Cconference on Pervasive Computing (PERVASIVE 2006), pp. 50–66
21. You C-W, Chen Y-C, Chiang J-R, Huang P, Chu H-H, Lau S-Y (2006) Sensor-enhanced mobility prediction for energy-efficient localization, in Proceedings of Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON 2006), pp. 565–574
22. Cheng YC, Chawathe Y, LaMarca A, Krumm J (2005) Accuracy Characterization for Metropolitan-scale Wi-Fi Localization, in Proceedings of the Third International Conference on Mobile Systems, Applications and Services (MobiSys), pp. 233–245
23. Krumm J, Hinckley K (2004) The NearMe Wireless Proximity Server, in Proceedings of the Sixth International Conference on Ubiquitous Computing, pp. 283–300

Arvin Wen Tsui is a Ph. D. student at the Graduate Institute of Networking and Multimedia, National Taiwan University and also works at the Industrial Technology Research Institute, Taiwan as a technical deputy manager. He received his B.S. (1994) in Computer Science from Soochow University and M.S. (1996) in Computer Science and Information Engineering from National Taiwan University. From 1996–1998, he was a research assistant at the Institute of Information Science, Academia Sinica. His current research interests include localization and ubiquitous computing.

Yu-Hsiang Chuang received the B.S. and M.S. degrees in information management from National Chung Chen University, Chiayi, Taiwan in 2004 and 2006, respectively. He is currently an engineer in the Information & Communications Research Laboratories of the Industrial Technology Research Institute of Taiwan, R.O.C. His research interests include localization over large-scale 802.11 wireless networks, and wireless sensor networks.

Hao-Hua Chu is an associate professor at the Graduate Institute of Networking and Multimedia and Department of Computer Science and Information Engineering, National Taiwan University. He received his B.S. (1994) in Computer Science from Cornell University and PhD (1999) in Computer Science from University of Illinois at Urbana Champaign. From 1999–2000, he was asenior software engineer at Intel. From 2000–2003, he was a project manager at (NTT) DoCoMo USA Labs. His research areas are pervasive computing and sensor/wireless network.