2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing

Rule-Based WiFi Localization Methods

Qiuxia Chen, Dik-Lun Lee
Computer Science and Engineering
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
{chen,dlee}@cse.ust.hk

Wang-Chien Lee
Computer Science and Engineering
The Pennsylvania State University
University Park, PA 16802, USA
wlee@cse.psu.edu

Abstract

The rule-based localization methods proposed in this paper are based on two important observations. First, although the absolute RSS values change with time, the relative RSS (RRSS) values between several Access Points (APs) are more stable than the absolute RSSs. Thus, we can use RRSSs as rules for inferring a client's location. Second, when a unique location cannot be obtained based on RRSS rules, the localization process can backtrack to the previous observed client location. By analyzing the accessible paths on the floor plan, locations that are not reacheable from the previous location can be disqualified. Based on these two key observations, we propose several localization methods, implement them in a life environment and conduct extensive experiments to measure the localization accuracy of the proposed methods. We found that our methods achieve much higher accuracy than the state-of-the-art localization methods, namely, RADAR, LOCADIO and WHAM!.

1 Introduction

The decreasing cost and proliferation of WiFi access points (APs) has made it very attractive to build indoor localization systems based on WiFi signal strengths. Most of them require an offline training phase to build a map overlaying the Received Signal Strength (RSS) space on the physical space, and an online localization phase to estimate the location by matching the measured RSSs on the client with the RSS map on the server.[1] Obviously, these methods do not perform well because RSSs change over time due to the dynamically changing environment.

The work proposed in this paper is based on two important observations. First, although the RSSs received from the APs at a location change over time, the *relative RSSs* (RRSSs), which refer to the relations of the RSSs received from the APs, are more stable than the absolute RSSs. For example, if the RSS from AP A is found to be larger than that of AP B, denoted by $RSS_A > RSS_B$, at a given location over a period of time, this relation tends to be quite stable even as the environment changes over time.[2] This is because the RSSs of the APs are often influenced by some static factors such as the distances, obstructions and the transmission powers of the APs, which tend not to change over time. Based on this observation, our methods define *rules* on RRSSs for every location in a space and infer a client's location by matching the rules observed at the client and the rules obtained for the space. We propose several rule-based methods and conduct experiments to measure their accuracy.

To validate this fundamental belief, we conduct an experiment in our experimental floor plan (see Figure 2) to measure the RSSs of AP2 and AP3 received in Rm 4208 from 8:00am to 14:00pm of a day. We found that of the 21,600 samples obtained, the relation $RSS_2 > RSS_3$ was satisfied 99.71% of the time. It shows that the relation is very stable despite environmental changes (being very quiet at 8am and very busy during lunch time). We will describe the details of how to derive the rules for a floor plan in this paper.

Our second observation is that many localization methods are based on a single set of RSSs measured at the client when localization is performed. Some methods made use of continuous monitoring of user locations to increase the localization accuracy [5]. Lee and Chen [8] proposed a backtracking technique to improve the accuracy of the localization results (see Section 2.2.1). In this paper, instead of using step-by-step backtracking, we collect RSSs periodically and use the Hidden Markov Model (HMM) and RRSS rules to derive the user's location.

The contributions of this paper are summarized below.

- We introduce the novel idea of using relative received

[1] Throughout this paper, we grade RSS values into 5 levels, from poor (0), fair (1), good (2), very good (3), to excellent (4).

[2] Since RSSs are graded into levels, it means that RSS of A is at a level higher than that of B.

978-0-7695-3492-3/08 $25.00 © 2008 IEEE
DOI 10.1109/EUC.2008.51

signal strengths (RRSSs) as the basis for localization, and developed and evaluated a suite of methods based on RRSSs.

- We introduce two types of RRSS rules for a floor plan, namely, dynamic rules and static rules, and, respectively, two classes of localization methods, namely, dynamic rule-based methods and static rule-based methods.
- An extensive experimental study is conducted to compare the accuracy of the proposed rule-based methods and three state-of-the-art WiFi localization methods, namely RADAR, LOCADIO and WHAM!. The experimental results demonstrate that rule-based methods outperform the other three methods significantly, in terms of accuracy.

The rest of this paper is organized as follows. Section 2 reviews the previous work on WiFi-based localization methods and the background of Hidden Markov Model. The RRSS rules are illustrated in Section 3, and the algorithms to manually and automatically deriving the rules for a floor plan are also presented. Section 4 describes six rule-based methods in detail. In Section 5, we evaluate the accuracy of our proposed methods and compare them with RADAR, LOCADIO, and WHAM!. The conclusion and some future research directions are presented in Section 6.

2 Preliminaries

In this section, we first review the related work and then provide a background on *the backtracking method* and *hidden markov model* used in our study.

2.1 Related Work

Indoor localization using WiFi signal strengths has attracted much attention in the ubiquitous computing community. The most well-known study is an in-building RF-based user location and tracking system developed by Microsoft Research (RADAR) [1]. In RADAR, WiFi RSSs were measured at various points of the test site. The readings were recorded in a server database. When a location query was submitted by the user, the system compared the current RSSs at the client with the RSSs in the server database. As a result, the most probable location could be derived and returned to the user. Several projects were conducted based on matching of WiFi RSSs [11]. Several statistical machine learning methods have been applied to WiFi localization [3, 9]. However, most of the previous works estimate a user's location based on matching the user's RSSs with the RSSs stored in the database, which were obtained at a previous time. Since RSSs change significantly over time, these systems are inaccurate and need to be retrained from time to time.

Most of the existing methods do not consider the continuous movement of a user, but some recent works made use of the user path and movement to enhance localization accuracy [5]. Lee and Chen [8] proposed an approach in which the client records RSSs periodically and disambiguates the user's location by backtracking to the user's previous locations. These methods take into account the continuity of locations when a user walks around. HMM-based methods [6, 7] utilize similar idea of continuity in user movement, but the techniques are very different from backtracking. They take into account the probability of seeing an AP at a given location and use a Hidden Markov Model (HMM) to identify the location of a user. A probability distribution map for RSSs from all APs is constructed offline for all locations. During the online localization stage, the system compares the RSSs from all APs with the map to calculate the corresponding probability for each location. Finally, the system reports the location with the highest probability.

Most, if not all, of these existing methods are based on the absolute RSSs. However WiFi signal strengths are unstable. Thus, it is hard to achieve good accuracy based on absolute RSSs. We show in this paper that RRSSs defined on pairs of APs provide a much more stable basis for localization. In this paper, we represent RRSSs as *rules* and develop rule-based methods for indoor localization.

2.2 Background

2.2.1 Backtracking

Lee and Chen [8] proposed a method called WHAM! (Where Am I), which is part of an indoor navigation and semantic location modeling project. In this approach, a client periodically records RSSs received from the APs. When localization is performed, the system will estimate the user's actual location using a standard WiFi localization method. If a unique location is obtained, the system will return the location to the user. When ambiguity arises, WHAM! will "backtrack" and estimate the client's immediately preceding location from its previously recorded RSSs.

By estimating the likelihood for the user to move from the previous location to the candidate locations, WHAM! can discard candidate locations that are not likely to be reached from the user's previous location. If a conclusive answer still cannot be reached, the system backtracks one more step until a clear answer is obtained or a system-defined backtracking level is exceeded, at which point a localization failure is reported. To facilitate disambiguation, this method requires a semantic location model to represent the locations and paths within the indoor environment [2, 4].

2.2.2 Hidden Markov Model

We review in this subsection the Hidden Markov Model (HMM) as introduced in [10]. A sequence of states at T successive times are taken into account and denoted as $W^T = \{w(1), ..., w(T)\}$ where the state at time t is denoted as $w(t)$. Notice that the system can visit any state at any step, but not every state needs to be visited. The mechanism producing the state sequence is the transition probability, denoted as $P(w_j(t+1)|w_i(t)) = a_{ij}$, which is the probability of reaching state w_j at time $t+1$ given that the state is w_i at time t. Note that the probability is time-independent. Hence, it can be denoted as a_{ij}, not $a_{ij}(t)$.

We continue to assume that at time t, the system in state $w_j(t)$ can emit some visible states, denoted as v_k. The number of the emitted visible states is T. The probability of emitting v_k can be formulated as $P(v_k(t)|w_j(t)) = b_{jk}$, where $1 \leq k \leq T$. In an HMM, we have access only to the visible states v_k's, while w_j's are unobservable.

In this paper, we require that some transition must occur from step $t \rightarrow t+1$ and some visible state must be emitted for every step. Thus, we have the following normalization conditions: (1) For any hidden state W_i, $\sum_j a_{ij} = 1$, and (2) for any hidden state W_j, $\sum_k b_{jk} = 1$.

From the above illustration, it can be seen that there are five necessary elements for a HMM: (1) the hidden states, (2) the visible states, (3) the transition probabilities among the hidden states a_{ij}, (4) the probabilities to observe the visible states at a given hidden state b_{jk}, and (5) the initial probabilities for all hidden states. The decoding problem in HMM can be described as follows. Given an HMM as well as a set of observations V^T, determine the most likely sequence of hidden states W^T leading to the observation V^T. In this paper, we make use of the decoding problem but omit the details due to the space limitation. Readers are referred to [10] for the detailed procedure to determine the most likely sequence of hidden states based on a sequence of observed states.

3 The Creation of Rules

As discussed earlier, RSSs change over time, making it difficult to obtain accurate results using an RSS map constructed at a particular time to estimate the user's location at a later time. As shown earlier, RRSSs from different APs at a given location remain very stable. In this section, we describe how to derive the rules.

Instead of using a geometric location model (i.e., the x and y coordinates of a user) for a floor plan, we aim to identify the semantic locations (e.g., rooms or corridors) of the user. We use a semantic location model to model a floor plan [4]. In the model, rooms and corridors are locations, which are connected by exits. Although a large room may be divided into several locations, we assume without loss of generality that a room corresponds to a single location. However, since a corridor could be very long, we allow a corridor to be divided into several locations so that a user walking on a corridor can be more precisely identified. Finally, a location represents at most one room or one corridor segment. Figure 1 shows a floor plan that consists of two rooms and a corridor connecting them. Consequently, each room creates a location and the corridor creates another location.

Next, we present how to 1) create static rules *manually*, and 2) create dynamic rules *automatically*.

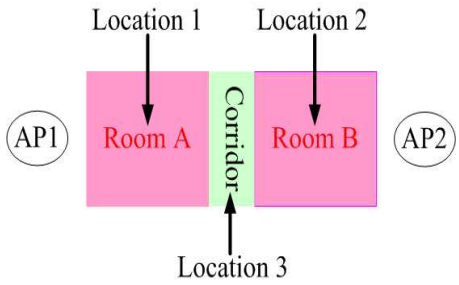


Figure 1. An example floor plan.

3.1 The Creation of Static Rules

The static rules for a location are created manually[3] based on the geometric properties of the floor plan. In the experimental floor plan, we employed several commonsense heuristics to create the static rules. For example, if a location is *much farther* from AP *x* than AP *y*, we write the rule $RSS_y > RSS_x$; in particular, if AP *x* is mounted inside a location, then $RSS_x > RSS_y$ for all y mounted outside the location. Furthermore, the number of turns is also taken into consideration. If the path from AP x has more turns than that of AP *y*, we write the rule $RSS_y > RSS_x$. Although we use RRSS extensively, we also include a small number of *absolute* rules based on RSS matching when and only when we found that the RSS from an AP is very stable (e.g., the AP is inside, or very near and in line-of-sight of the location).

Taking the floor plan in Figure 1 as an example, there are three locations, l_1, l_2, l_3 corresponding to Room A, Room B and the corridor, respectively. Intuitively, we can construct three rules $rule_1 = (RSS_1 > RSS_2)$, $rule_2 = (RSS_1 < RSS_2)$ and $rule_3 = (RSS_1 = RSS_2)$ based on the layout of the floor plan and the locations of the APs in the floor plan to identify, respectively, l_1, l_2, and l_3. In addition, a new rule $rule_4 = (RSS_1 = 4)$ can be defined to apply only

[3]It is conceivable that status rules can be created by automatic reasoning on the floor plan, but this is not the focus of the paper.

to l_1. Multiple rules can be used to improve the accuracy of localization, especially in dynamic environments.

3.2 The Creation of Dynamic Rules

Dynamic rules are created by measuring RSSs of each AP in each location and then generate the rules automatically. First, we build an average RSS table to record the average RSS from all APs for each location in a day. Then, for each location, rules are generated for each pair of APs based on the average RSSs of the APs at that location. Measuring the average RSS values over a day ensures that the resulting RRSS rules are not affected by transient interferences during the measurement period. Furthermore, APs that have low RSSs are not used in generating the rules.

Suppose the total number of APs is N in the floor plan. The number of all possible pairs of APs is C_2^N. Given a pair (RSS_i, RSS_j), three possible rules can be obtained, i.e., $RSS_i > RSS_j$, $RSS_i = RSS_j$ and $RSS_i < RSS_j$. Therefore, the total number of all possible rules is $3 * C_2^N$ for each location.

In general, different locations result in different rules. However, it is still possible that two different locations l_i and l_j have the same rule, i.e., $r_i = r_j$ (e.g., when they are very close together). Further, a rule can be a subset of another rule, i.e., the RSS relations in one rule is a subset of the RSS relations of another rule. In both cases, matching of one rule implies the match of another rule, thus returning more than one location.

4 Rule-Based Localization Methods

We use the prefix DR and SR to denote rule-based localization methods that are based on static and dynamic rules, respectively. In general, most of the previous methods can be modified to use RRSS rules instead of absolute RSSs to enhance accuracy. In this paper, we extended the backtracking method proposed by Lee and Chen [8] to use RRSS rules instead of absolute RSSs. The extended methods, to be discussed in detail later, are called dynamic and static rule-based backtracking localization methods (DRBTM and SRBTM, respectively). In addition, since HMM works well in dealing with a sequence of signals, we also apply RRSS rules as input to HMM to develop the dynamic and static rule-based HMM localization methods (DRHMM and SRHMM, respectively).

In the following, we present only SRLM, SRBTM and SRHMM in detail since their main differences from their dynamic counterparts, namely, DRLM, DRBTM, and DRHMM, are only on the different types of rules adopted.

4.1 Static Rule-Based Localization Method (SRLM)

Given a floor plan, the semantic locations and rules corresponding to all locations can be obtained. When a user moves around and submits a query to the system, the system will record and pre-process the RSSs for different locations. The system compares the pre-processed RSSs against the rules stored in the server in order to determine their locations.

4.2 Static Rule-Based Backtracking Method (SRBTM)

It is obvious that multiple locations may be returned by DR or SR. As shown in [8], backtracking resolves the ambiguity problem and enhance the accuracy of localization using the user's movement path. The first step of SRBTM is the same as that of SR. After a user submits a location query, the system will report a list of possible locations to the user. By backtracking and considering the connection between locations in the floor plan, a more accurate estimation can be deduced [8].

4.3 Static Rule-Based HMM Method (SRHMM)

As presented above, we use the user's movement path to enhance the localization accuracy. The idea is to identify a unique location by considering the connectivity relation of two neighbor locations. Although backtracking is an attempt to exploit the same idea, it considers the connectivity step by step and as such the global connectivity is not considered. In HMM-based methods, the sequence of signals can be considered as a decoding problem in HMM. Thus, we apply RRSS rules to HMM to develop rule-based-HMM localization methods. We model an HMM as follows.

- Hidden States: the semantic locations are defined as hidden states.
- Visible States: the rules are defined as visible states.
- The Transition Probabilities among hidden states a_{ij} will be calculated by the physical correlations of locations. For example, location l_i are connected to p locations including itself l_i and l_j, thus $a_{ij} = 1/p$.
- The probabilities to observe visible states at a given hidden state b_{jk}. For a given location l_j, q rules, $r_j = \{Rule_{j1}, \ldots, Rule_{jq}\}$, are obtained. Without loss of generality, $rule_k$ is included in these q rules. Thus $b_{jk} = 1/q$.

- Initial probabilities for all hidden states. We set an equal value to each probability of any hidden state for lack of any initial information.

Based on the above HMM, a sequence of visible states is submitted when a user submits a location query. By solving the decoding problem, the system will determine a sequence of locations with the largest probability.

5 Experimental Evaluation

In this section, the effectiveness of our rule-based methods is evaluated through extensive experiments. A comparison with three state-of-the-art localization methods, RADAR [1], LOCADIO [6], and WHAM! [8], are conducted. We implemented the methods and evaluated their accuracy in an area around our research labs. Figure 2 shows the floor plan used in our experiments, including the locations of the APs and the boundaries of the semantic locations. Note that each room corresponds to a location. However, long corridors (colored blue in the figure) are divided into several locations to make them more precise. The floor plan has a dimension of $43.5m$ by $40.5m$. In our experiments, a person carries an IBM laptop equipped with an Intel(R) Pro/Wireless 2200BG card and performs tests at different locations in this testing environment.

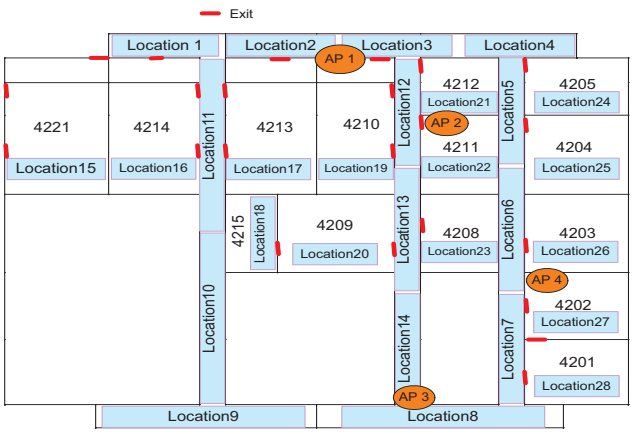


Figure 2. Floor plan for experiments.

5.1 Performance Measures

Since some methods can return multiple locations for a query, which may or may not include the correct location, we evaluate the methods under both settings to facility comparison and understanding of the methods. First, when every method is allowed to return only one result, the accuracy is measured by:

$$R_u = \frac{\textit{Number of queries returning the unique correct results}}{\textit{Total number of queries submitted}} \quad (1)$$

Second, when multiple candidate results are allowed to be returned, the accuracy is measured by:

$$R_m = \frac{\textit{Number of queries returning the correct results among the returned candidates}}{\textit{Total number of queries submitted}} \quad (2)$$

It is clear that high R_u is much more important to users than R_m. However, R_m provides one more performance measure for us to consider if for some reasons some methods cannot return the unique correct results we may still want to know which methods perform better by considering as better those that are able to return the correct results among the candidates.

We also measure the *error distance* (denoted as E_d), which is the length of, measured in terms of the number of locations passed through by, the shortest path between the returned location and the correct location. For example, the error distance between location 1 and location 2 in Figure 1 is 2 because location 3 is located between them.

5.2 Rule Creation

For the rule-based methods, we first create dynamic and static rules for all locations. Due to space limitation, we only show 12 static rules out of 35 static rules in Table 1. The static rules are created based on the geometric distances from the locations to the APs and fine-tuned with the RSSs received from the APs at a certain time during the evaluation.

After the rules are derived, DRLM, SRLM, DRBTM, and SRBTM are ready to be evaluated. However, for DRHMM and SRHMM, we set the probabilities of the hidden states to the same value for the lack of any initial information.

In order to demonstrate the effectiveness of our rule-based methods, we test the correctness of localization assuming that the user is moving along a paths (termed as *path test*).

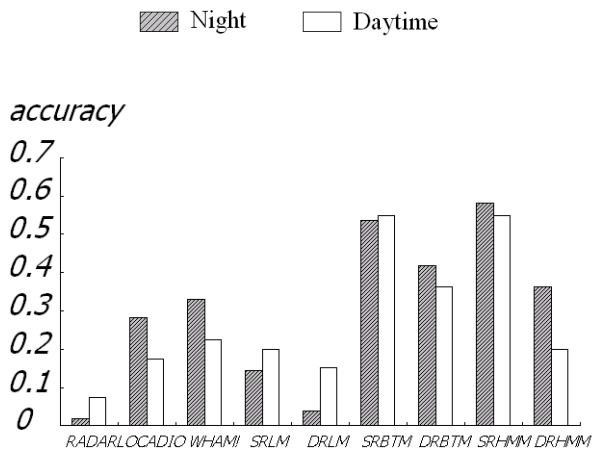
5.3 Effectiveness of Localization in Path Tests

For the path test, we randomly select 16 paths of various lengths in the testing area. We measure the performance of all methods during daytime and at night. A user with a laptop computer walks through each path and submits a location query to the system at the end of the path. The laptop samples a set of RSSs received from the APs once

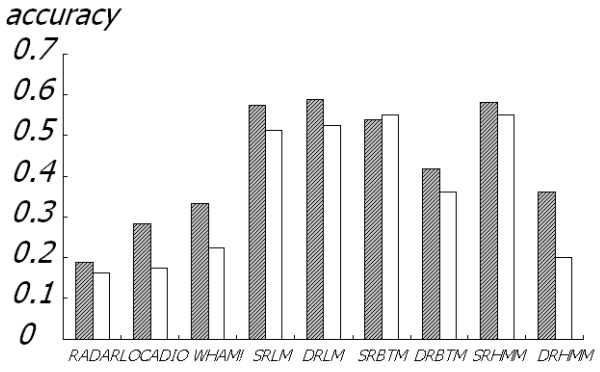
Rules No.	Location	Part1	Part2	Part3	Part4	Part5	Part6
1	1	$RSS_1 = 2$	$RSS_2 = 1$	-	-	-	-
2	2	$RSS_2 >= RSS_1$	$RSS_2 < 3$	-	-	-	-
3	9	$RSS_1 < 3$	$RSS_2 = 0$	-	-	-	-
4	10	$RSS_1 = 3$	$RSS_2 = 1$	-	-	-	-
5	11	$RSS_1 >= RSS_2$	$RSS_1 = 3$	$RSS_2 > 1$	$RSS_3 > 0$	-	-
6	15	$RSS_1 > RSS_2$	$RSS_1 > 2$	$RSS_3 < 2$	-	-	-
7	16	$RSS_1 > RSS_2$	$RSS_1 = 4$	-	-	-	-
8	17	$RSS_1 <= RSS_2$	$RSS_2 > 2$	-	-	-	-
9	18	$RSS_1 > 1$	$RSS_1 < 4$	$RSS_2 > 1$	$RSS_3 > 0$	$RSS_1 >= RSS_2$	$RSS_1 >= RSS_3$
10	2	$RSS_2 > RSS_3$	$RSS_2 = 2$	$RSS_4 = 0$	-	-	-
11	3	$RSS_2 = RSS_3$	$RSS_2 = 2$	$RSS_4 = 0$	$RSS_1 = 0$	-	-
12	4	$RSS_1 = 0$	$RSS_4 = 0$	$RSS_2 > 1$	$RSS_3 > 1$	-	-

Table 1. The static rules

every two seconds during the walk and store them on its local storage. The walking speed is about 3 feet per second.



(a) R_u



(b) R_m

Figure 3. Ratio of correct results for path test.

From Figure 3(a), it is clear that although LOCADIO and WHAM! outperform DRLM and SRLM, most of the rule-based methods have better R_u than RADAR, LOCADIO and WHAM! for both daytime and night time. For example, during daytime, SRHMM has R_u of 58%, while WHAM! has R_u of 33%, LOCADIO 28%, and RADIO only 2%, which is the worst. Furthermore, at night time, the rule-based methods still have higher accuracy. The best ones are SRBTM and SRHMM.

Figure 3(b) shows the R_m of path test for all of the methods, while Figure 4 compares R_u and R_m in path test. The performance improves significantly for SR and DR, i.e., they are able to return results that include the correct locations. However, this "improvement" should be taken as a reference only since in reality returning several candidate locations is of little value to the user even though the correct location is included in the result. The performance of LOCADIO, SRHMM and DRHMM have no difference since these methods do not return multiple results. For WHAM!, SRBTM, and DRBTM, ambiguous results have been resolved using backtracking method.

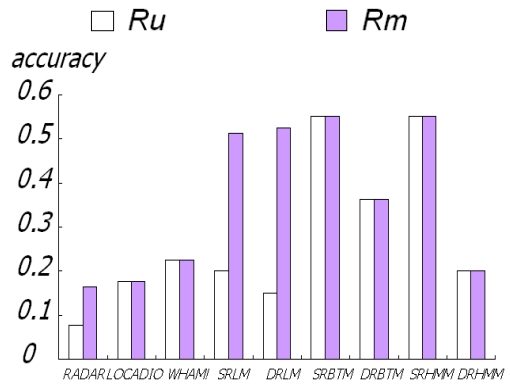


Figure 4. Comparison of R_u and R_m in path test during daytime.

The E_d values for the methods are shown in Figure 5. It is clear that although WHAM! performs well, most of the rule-based methods work better than RADAR and LOCADIO in terms of error distance. DRHMM has the smallest average error distance at 1.78 (number of locations from the actual location).

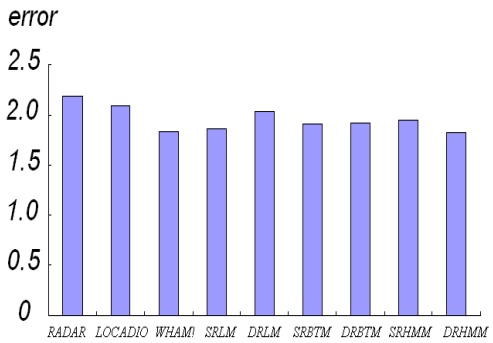


Figure 5. Average error distance (E_d) for path test.

In conclusion, our proposed rule-based methods outperform RADIO, LOCADIO and WHAM! in path tests. The rule-based backtracking methods and rule-based HMM methods are more suitable for localization when the user is walking around.

6 Conclusions and Future Work

In this paper, we present a number of rule-based indoor localization methods. The primary idea behind the proposed rule-based methods is that relative RSS (RRSS) from different APs are more stable than absolute RSS in dynamic environments. Through extensive empirical experiments, we validated this idea and demonstrated that the methods proposed in this study significantly outperform the state-of-the-art WiFi localization methods. For future research, we would like to further employ our idea to develop new approaches for WiFi localization and to enhance our methods by facilitating automatic updates of rules when APs are added, removed or relocated.

Acknowledgment

Qiuxia Chen and Dik Lun Lee were supported by the Research Grant Council, Hong Kong SAR, through Grants 615707, 615806, 616005 and CA05/06.EG03. Wang-Chien Lee was supported in part by the National Science Foundation under Grant no. IIS-0534343 and CNS-0626709.

References

[1] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. *INFOCOM 2000: Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2:775–784 vol.2, 2000.

[2] F. Dürr and K. Rothermel. On a location model for fine-grained geocast. In *UbiComp '03: Proceedings of the 5th International Conference on Ubiquitous Computing*, pages 18–35, 2003.

[3] B. Ferris, D. Fox, and N. Lawrence. Wifi-slam using gaussian process latent variable models. In M. M. Veloso, editor, *IJCAI*, pages 2480–2485, 2007.

[4] H. Hu and D. Lee. Semantic location modeling for location navigation in mobile environment. In *MDM '04: Proceedings of the 5th international conference on Mobile data and management*, pages 52–61, Berkeley, USA, 2004.

[5] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. In *WMASH '04: Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots*, pages 110–118, New York, NY, USA, 2004. ACM.

[6] J. Krumm and E. Horvitz. Locadio: inferring motion and location from wi-fi signal strengths. In *Mobiquitous '04: First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, Boston, MA, USA, 2004.

[7] A. M. Ladd, K. E. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. S. Wallach. Robotics-based location sensing using wireless ethernet. In *MobiCom '02: Proceedings of the 8th annual international conference on Mobile computing and networking*, pages 227–238, New York, NY, USA, 2002. ACM.

[8] D. L. Lee and Q. Chen. A model-based wifi localization method. In *Infoscale '07: The Third International ICST Conference on Scalable Information Systems*. ACM, 2007.

[9] J. J. Pan, Q. Yang, H. Chang, and D.-Y. Yeung. A manifold regularization approach to calibration reduction for sensor-network based tracking. In *AAAI*, 2006.

[10] P. E. H. Richard O. Duda and D. G. Stork. Pattern classification, second edition. In *John Wiley and Sons, Inc*, New Jersey , NJ, USA, 2001.

[11] Z. Xiang, S. Song, J. Chen, H. Wang, J. Huang, and X. Gao. A wireless lan-based indoor positioning technology. *IBM J. Res. Dev.*, 48(5/6):617–626, 2004.