WiFi iLocate: WiFi based Indoor Localization for Smartphone

Xiang He, Shirin Badiei, Daniel Aloi, Jia Li
Electrical and Computer Engineering
Oakland University, OU
Rochester, MI 48309, U.S.A
{xhe2, sbadiei, aloi, li4}@oakland.edu

***Abstract*—In recent years, the increasing popularity of smartphones has promoted the development of location-aware applications. However, highly accurate indoor localization by smartphones remains an open problem. In this paper, we present WiFi iLocate – a system that can help track the location and movement of a smartphone user in indoor environments. The system applies Gaussian process regression to train the collected WiFi received signal strength (RSS) dataset, and particle filter for the estimation of the smartphone user's location and movement. Simulations were conducted in MATLAB to test the performance and provide more insights of the proposed approach. The experiments carried with an iOS device in typical library environment illustrate that our system is an accurate, real-time, press-to-go system.**

***Keywords*—WiFi RSS, indoor localization, Gaussian process regression, particle filter, smartphone**

I. INTRODUCTION

Nowadays, WiFi access points (APs) have become ubiquitous, whether in the offices, museums, shopping malls or airports. In the mean time, smartphones are playing more and more important roles in people's daily life. We often see people with smartphones walking around in the public areas. An accurate indoor localization system can help people easily accessing navigation in a museum or airport terminal, finding specific merchandise or promotion information in a shopping mall, or locating themselves whenever they get lost. Global Positioning System (GPS) is commonly used for navigation outdoors. But it lacks enough accuracy when functioning in indoor environment. People are trying to develop WiFi based Positioning System (WPS) to fulfill the indoor localization task.

A WiFi based localization system has several advantages: First, WiFi APs are becoming ubiquitous in many of the indoor environments. Second, WPS only rely on the existing infrastructure, so no modification to the environment is required. Third, the WiFi information needed for doing localization include only the received signal strength (RSS) and the Basic Service Set Identifier (BSSID). The information is easy to collect, simply by sniffing the wireless traffic in the air. Fourth, WiFi signals do not require line-of-sight (LOS). It is extremely suitable for the use in indoor environment where there are a lot of walls and obstructions.

Generally speaking, WiFi based indoor localization techniques can be categorized into two types, propagation based and location fingerprinting based. Propagation based algorithms usually apply mathematical models to a set of triangulation algorithms to determine the location of the device. The triangulation approach uses the geometric properties of triangles to estimate the target location. Specific techniques use information include angle of arrival (AOA), phase of arrival (POA), time of arrival (TOA) and roundtrip time of flight (RTOF) to localize the device [1]. The main drawback of the propagation based algorithms is the difficulty in getting an accurate propagation model for the complicated indoor environment. Due to this difficulty, propagation based techniques can only achieve limited accuracy.

Instead of modeling the propagation of the WiFi signal, location fingerprinting based algorithms assumes that a WiFi enabled device always receives similar signal strength at a certain location, such RSS and coordinates would serve as a unique "fingerprint" of this location. We can collect a "fingerprint" in each location and store them in a dataset. Every time we come to a new location, the Wi-Fi signal strength is detected and the location is estimated by measuring the similarity between current and stored fingerprints.

There are many location fingerprinting based methods, such as the K-Nearest Neighbor (K-NN), neural networks, support vector machine (SVM) and the probabilistic method [1]. The K-NN algorithm compares the online scanned WiFi RSS with the offline built WiFi RSS dataset, searches for K closest matches RSS values in signal space, and uses these K known locations to estimate the current location. K-NN is easy to implement but it suffers greatly from signal fluctuations as the RSS detected at the same location may vary from time to time. Neural Network and SVM both are machine learning methods used for classification and regressions. They require huge amount of training data and complex training process to achieve high accuracy result, which are not applicable in smartphone. By introducing Gaussian process, we are able to build an accurate WiFi RSS model without the need of intensive survey data collection and sophisticated training process.

Probabilistic method finds the most probable location out of the pre-recorded location fingerprinting dataset [1]. During the online localization phase, the likelihood of different location candidates can be obtained from the observation of scanned WiFi RSS, and the location estimation is calculated using

978-1-4799-1297-1/14/$31.00

Bayesian filtering technique. By implementing Bayesian filtering through a particle filter, we build a localizer makes use of current observation and previous location information to estimates the posterior location. It is most suitable for real time localization and tracking of a smartphone, we will discuss this method in detail in Section II.

Next, we would like to mention about the implementation issue. Extensive research has been done in developing the indoor localization systems, such as RADAR [2], Horus [3], Compass [4]. However, most of them are developed on laptop platforms equipped with better antennas than on the smartphone. Moreover, recent work on developing smartphone indoor localization app achieves only room level accuracies, like Shopkick[5].

Therefore, accurate indoor localization on smartphone still remains an open problem. Liu et al [6] tried to solve this problem by introducing peer assisted localization approach. But this approach only works in public areas with high densities of smartphones present at the same time. Lokesh et al [7] described an accurate smartphone based indoor pedestrian localization system using WiFi and camera on the phone. But they only demonstrated their results in simulation instead of an actual implementation on smartphone.

Several RSS based localization systems that utilize Gaussian process regression have already been developed, and this approach has proved to be well-suited in modeling the RSS dataset [8] [9] [10]. But none of them is adapted to the smartphone platform. Our approach is inspired by their work and tailored for the purpose of smartphone application. The two main components of our WiFi based localization algorithm are a WiFi RSS dataset trained by Gaussian process regression and a localizer based on particle filter.

Specifically, we make the following contributions: We built a WiFi based indoor localization and tracking system on an iOS platform. It does not require any dedicated infrastructure in the indoor area or specialized hardware equipped on the smartphone. Moreover, our system does not require any user-specific information, such as user's initial location or AP's location. It is press-to-go localization. To the best of our knowledge, our iOS application WiFi iLocate is the first one to achieve real time, highly accurate indoor localization by leveraging Gaussian process WiFi RSS fingerprinting modeling along with particle filter based localizer in smartphones.

The remainder of this paper is organized as follows: In Section II, we give an overview of Gaussian process regression and show how it can be used in modeling WiFi RSS fingerprinting. Then we describe a localizer using particle filter to do the location estimation based on this model. In Section III, simulation is conducted to prove the feasibility of the proposed localization algorithm, and provide us with more insights of the algorithm. The iOS implementation, named WiFi iLocate, and its performance in real situation, are presented in Section IV. Finally we conclude the paper in Section V with a discussion of future research direction.

II. SYSTEM SETUP

The probabilistic location fingerprinting method uses Bayesian filtering to determine the location under estimation [12]. Let $p(x_i \mid z)$ denotes the probability that the WiFi enabled device is in location x_i given the received signal vector is z. We select a location x_i if $p(x_i \mid z) > p(x_j \mid z)$, for i, j = 1, 2, 3, ... n, $i \neq j$.

Also let's assume that $p(x_i)$ is the probability that the smartphone is in location x_i, $p(z \mid x_i)$ is the probability that the signal vector z is received, given that the device is located in location x_i. The given decision rule is based on posterior probability

$$p(x_i \mid z) = \frac{p(z \mid x_i)p(x_i)}{p(z)} \tag{1}$$

Here we can assume that $p(z)$ is a constant for all x, so that the formula can be rewritten as

$$p(x_i \mid z) \propto p(z \mid x_i)p(x_i) \tag{2}$$

The estimated location x is the one obtains the maximum value of the probability

$$x = \arg\max_{x_i}[p(x_i \mid z)] \tag{3}$$

Traditionally, we collect the location fingerprints on the offline training phase and create a dataset to store them. During the online localization phase, we calculate the likelihood $p(z \mid x_i)$ of each location candidate based on the observed signal strength, and the estimated location can be decided by the Bayesian decision rule discussed above.

However, we can only collect fingerprints in discrete location, which makes this technique only applicable for discrete location estimation. On the other hand, smartphone user can be at any location, and move in a continuous manner. Therefore, we need to interpolate through the collected fingerprints. Gaussian process provides us such an advanced interpolation method.

A. Gaussian process regression for WiFi signal strength modeling

Gaussian processes (GPs) offer many advantages that make them suited for a localization system that utilizes WiFi signal strength [8]:

Firstly, GPs are non-parametric, a mathematic model that can correctly fit the data is not required. Because GPs place a prior over the distribution of functions, many highly non-linear models can emerge from GP regression. Here we use GPs to approximately fit the non-linear WiFi signal propagation model.

Secondly, GPs do not require a discretized representation of an environment, or the collection of calibration data at pre-specified locations. They can predict signal strength measurements at arbitrary locations.

Thirdly, GPs provide uncertainty estimation for predictions at any given locations. This uncertainty is measured in variance, which takes into account the training data density and the noise of the data.

A Gaussian processes essentially estimates a posterior probability distribution over functions from training data. The details on GPs can be found in [11]. We will give a brief introduction here.

Let's first define a function $f(x_*)$ be the posterior distribution that makes prediction for all possible input x_*. And we have $D = \{(x_i, y_i) \mid i = 1,...,n\}$, which is a set of training samples consists of n observations drawing from a noisy process $y_i = f(x_i + \varepsilon)$, where each x_i is an input sample in $\Re^d$ and each y_i is a target value in $\Re$, ε is additive Gaussian noise with zero mean and variance σ_n^2. For notational convenience, the inputs of the training set are grouped into a $d \times n$ matrix X, and the observations y_i are grouped into a vector y.

To estimate the posterior distribution over function $f(x_*)$ from training dataset D, GPs depend on a covariance function kernel $k(x_p, x_q)$, which specifies how the values at different points are correlated to each other. This kernel can be specified as any arbitrary covariance function, and we have chosen the widely used squared exponential kernel

$$k(x_p, x_q) = \sigma_f^2 \exp(-\frac{1}{2l^2} \mid x_p - x_q \mid^2) \qquad (4)$$

Here, The hyperparameters σ_f^2 and l are the signal variance and the length scale which determines the strength of the correlation between different points.

Since we only have access to the noisy observations y instead of the true function value $f(x)$, we must add a term to account for observation noise in the covariance function:

$$\mathrm{cov}(y_p, y_q) = k(x_p, x_q) + \sigma_n^2 \delta_{pq} \qquad (5)$$

Here σ_n^2 is the Gaussian observation noise and δ_{pq} is one if $p = q$ and zero otherwise. For an entire set of input values X, the covariance over the corresponding observations y can be written as

$$\mathrm{cov}(y) = K + \sigma_n^2 I, \qquad (6)$$

where K is the $n \times n$ covariance matrix of the input values, defined as $K[p,q] = k(x_p, x_q)$.

Note that the covariance between the observations is written as a function of the inputs, emphasizing the non-parametric nature of Gaussian process regression.

Now we can generate the posterior distribution over functions $p(f(x_*) \mid x_*, X, y) \sim N(\mu_{x_*}, \sigma_{x_*}^2)$ to predict the function value for any arbitrary points x_*, given the training data X and y:

The prediction's mean and variance are:

$$\mu_{x_*} = k_*^T (K + \sigma_n^2 I)^{-1} y \qquad (7)$$

$$\sigma_{x_*}^2 = k(x_*, x_*) - k_*^T (K + \sigma_n^2 I)^{-1} k_* \qquad (8)$$

The hyperparameters σ_f^2, σ_n^2 and l control the smoothness of the functions estimated by a GP and can be learned from training data, by maximizing the log marginal likelihood of the observations conditioned on the hyperparameters. This learning process is completed offline right after the training dataset is built.

To apply GP in WiFi signal strength modeling, the input values X correspond to locations, and the observations y correspond to signal strength measurements gathered at these locations. The GP posterior is estimated from a collection of signal strength measurements corresponded with their locations. Assuming independence between different APs, we estimate a GP for each AP separately. After the generation of WiFi signal strength model, we can move to discuss the online particle filter based location tracking.

B. Location Estimation Based on Particle Filter

After the offline training phase to generate the GP based WiFi signal strength model, we are ready to use it in online localization.

During the online localization phase, we are going to determine the smartphone location using Bayesian filtering technique, implemented through a particle filter. Particle filter is able to handle any arbitrary probability density function. It has been adopted by several researchers in location estimation problem and has showed its advantages [8] [9].

As discussed before, the Bayesian filtering is based on formula (2).

Here $p(z \mid x_i)$ and $p(x_i)$ represent a measurement likelihood model and a motion model, respectively.

The measurement likelihood model can be calculated using the posterior distribution of the signal strength at each location determined by the GP

$$p(z \mid x_*) \propto \frac{1}{\sqrt{2\pi\sigma_{x_*}^2}} \exp\left(-\frac{(z-\mu_{x_*})^2}{2\sigma_{x_*}^2}\right), \qquad (9)$$

where μ_{x_*} and $\sigma_{x_*}^2$ are the posterior mean and variance at an arbitrary location x_*.

Location estimation algorithm using particle filter is performed according to the following steps:

(1) Particle Initialization

In order to realize location estimation in real-time using a smartphone, the computational complexity needs to be restrained. For particle filter, the computational complexity depends on the number of particles, N particles takes $O(N)$ time. We have tested different number of particles in our simulation to see how this number will affect the performance.

The initial location is calculated through weighted K nearest neighbor (W-KNN) method. It searches for K closest matches of known locations in signal space from the offline-built dataset. By averaging these K location candidates with adopting the distances in signal space as weights, the initial estimated location is acquired. This initial location estimation is used as the starting point for particles. We assume that the accuracy of the initial guess will not affect the localization performance. Larger initial error will only cause longer time for particles to converge to the actual location of the smartphone.

(2) Particle Movement

Next, the particles' coordinates are updated as each particle move. We choose not to use motion information under the assumption that for our WiFi localization system, this information may not be available or too noisy to be used. Therefore, in order to keep our location estimation only based on WiFi signals, the motion model is replaced with random particle movement. That is, in every time step, the particle cloud is spreading in all directions and can reach random distance within a reasonable range.

(3) Weight updating

When the particles start moving, the collected WiFi received signal strength (RSS) continually changes the likelihood of all particles. For each particle, the predicted signal strength mean and variance are calculated from the GP model. They are used to calculate the corresponding particle's likelihood $p(z \mid x_i)$.

In every update, the likelihoods are calculated for each AP. Here we assume that the APs are independent of each other, so that we compute the likelihood of a complete set of readings from all the APs by multiplying the individual reading likelihoods together. This combined likelihood is then treated as the weight for particle. When all the particles' weights have been calculated, normalization is performed so that the sum of all the particle weights equals one.

(4) Resampling and location estimation

After particle weights are updated, we perform importance resampling to update the particles' location. In resampling, the weight of each particle is treated as a probability where this particular particle is chosen to be the estimated location. Those particles with higher weights will be picked more frequently than others. This is how the resampling is able to eliminate those wrongly moved particles and correctly track the smartphone's location. After the resampling process, the estimated location is calculated as the mean of all the resampled particles' location.

III. Simulation and Result Analysis

We have validated the proposed WiFi based localization algorithm through simulation. The simulation and result analysis is performed in MATLAB.

Wireless InSite, an EM solver by REMCOM, has been used to simulate a 40m by 40 m empty room with 4 APs, as shown in Figure 1. Mesh grid with grid size of 0.2 m is used to sample the WiFi RSS. A total of 40000 RSS data points are collected for each AP. We pick 400 equally spaced points out of the total as survey points to train the GP model. After the offline training process, a random path is generated in the room to see if the online localization algorithm could catch the actual path, and estimate the error between the true path and the estimated path.

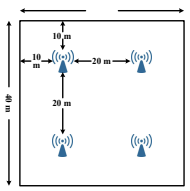

Figure 1: Simulated environment

We import all the WiFi RSS data into MATLAB, and demonstrate the online localization in a small animation in MATLAB. As shown in Figure 2, this true path, shown in red circle, starts at coordinate (4.5, 4.5) and ends at coordinate (25.5, 10.5). For simplicity, the path is on the grid with equal gap between each step. On the right, it shows a one time simulation result. The particle distribution corresponds to different locations are shown in blue dots and the estimated path is shown in red stars. We have set the particle number to 1000 and use all 4 APs in this simulation.

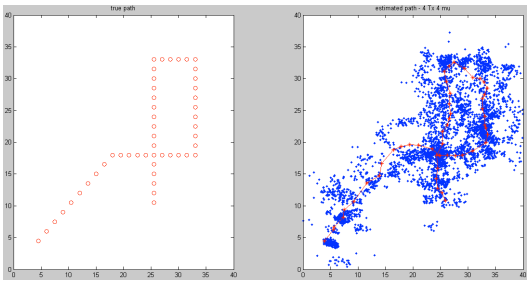

Figure 2: Particle filter estimation of a random path

To evaluate the accuracy of the localization algorithm, the simulation has been performed 100 times and the estimated location is compared to the ground truth location. The error is measured as the Euclidean distance between the actual location shown in red circle and the estimated location shown in red star. The average error in Figure 2 is 3.8 m.

Furthermore, we analyze the system performance under various situations.

Firstly, we explore how the number of AP can affect the location accuracy estimation. We test the AP number from 1 to 4, and fix particle number to 1000. The result is showed in Table 1.

AP Num	1	2	3	4
error	14.8 m	7.4 m	6.5 m	3.8 m

Table 1: The average error for different number of AP

As can be seen in Table 1, when the number of AP increases, the error drops significantly. This is reasonable as more APs provide more information and can achieve better location estimation accuracy.

Secondly, we investigate how the initial accuracy will affect the localization performance. We test the initial error from 0.5 m to 3 m, with particle number and AP number set to 1000 and 4, respectively. The result is shown in Figure 3.

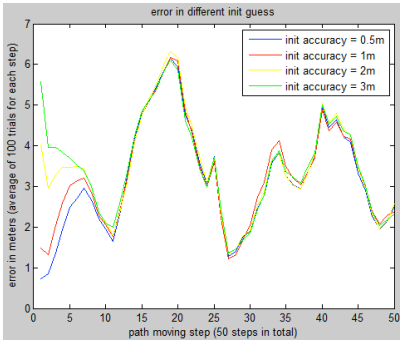

Figure 3: Localization performance in different initial error

Figure 3 shows that the algorithm is not sensitive to the accuracy of the initialization. Better initial accuracy only gives a better performance in the beginning few steps. After 8 steps, the particle filter compensates the difference and gives a similar performance onwards. This result verifies our hypothesis in the particle initialization step, that particle initialization will not affect the location estimation performance as they will finally converge to the ground truth location.

Thirdly, it is desirable to know how many particles are needed for an accurate localization. The number of particles from 200 to 2000 is tested, with AP number equals 4. The result is shown in Table 2.

particles	200	400	1000	2000
error	4.6 m	4.2 m	3.8 m	3.7 m

Table 2: The average error for different number of particles

As seen from Table 2, by increasing the number of particles, we can improve the localization performance. But using more particles means higher computational cost at the online localization phase. The benefit of accuracy improvement is very little when the particle number reaches a large enough value, as this simulation unveils, around 1000. This is a reasonable particle number to obtain good localization accuracy and real time response.

Lastly, we are interested in how the extra sensors, like accelerometer, gyroscope and compass could improve the localization accuracy, as our target devices - smartphones, do have these common sensors equipped.

A simple motion model was built based on the information provided by motion sensors to constraint the particles' movement. This motion model guides the particles to certain direction range and limits the step length to a reasonable range.

Figure 4 shows an estimated path with the help of motion model. We can observe that the particle distribution is more concentrated around the true path as compared to Figure 2.

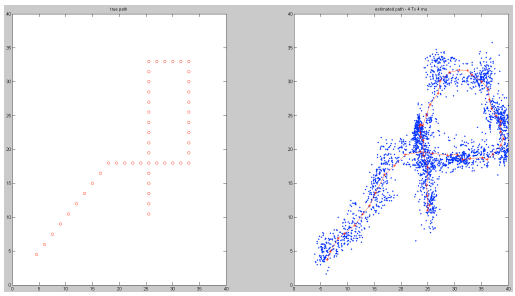

Figure 4: Particle filter location estimation with motion model

Error	No motion model	With motion model
1 AP	14.8 m	5.5 m
4 AP	3.8 m	2.9 m

Table 3: Localization accuracy between WiFi iLocate against weighted KNN

Table 3 shows the comparison result of no motion model against with motion model under different number of AP. It can be seen that the simple motion model greatly increases the location estimation accuracy. When we don't have many APs to provide enough information of the WiFi signal strength, we will need to use the motion model to guide the particle movement. Even with enough AP, the performance still improves by about 1 m in location estimation accuracy with the motion model equipped.

IV. Implementation in iOS Device

We realize the WiFi indoor localization algorithm on the iOS platform and built an app called WiFi iLocate. The system performance has been tested on the 3rd floor in Oakland University library.

A. System architecture

Figure 5 presents the workflow of WiFi iLocate. In brief, we first import the floor plan into the system. With the floor

plan display on the screen, we can set the survey points and scan WiFi. The scanned RSS values and corresponding BSSIDs are stored in an offline training dataset. After the construction of WiFi RSS dataset and preprocessing, we are able to perform the online localization by pressing the "Locate" button. Both the offline and online phases are completed on the iOS platform.

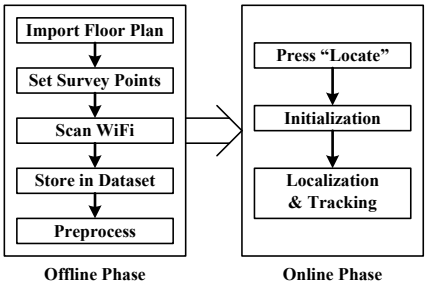

Figure 5: WiFi iLocate workflow

One important issue for WiFi scanning is that APs may be missed in a scanning cycle both during the offline surveying and the online localization. To overcome this issue, we perform multiple scanning in offline WiFi surveying and apply GP modeling to deal with the missing alue. During the online localization, interval is set as 4 seconds, which means we update our location estimation every 4 seconds. We scan once every period, compare the scanned BSSIDs with pre-store BSSIDs in the dataset, and use all the detected APs to calculate the current particles' weights.

The most computational costly step to generate the posterior of GP lies in the inverse of the covariance matrix: $(K+\sigma_n^2 I)^{-1}$, which takes time $O(N^3)$, where N is the number of survey points. Fortunately, this computation can be done before the real time localization step. After the creation of the training dataset, preprocess is performed and this inverse covariance matrix is stored in memory beforehand.

Right after we press the 'Locate' button, the iOS device starts scanning the WiFi RSS from all the APs it can detect. Particles are initialized through weighted KNN method. The number of particles is set to 1000 according to the simulation result. After particles are generated from the initial location, they start to move randomly, every 4 seconds the particles are resampled, as discussed in Section II. Through particle filtering, we can locate the user and track the user movement in real time.

B. Experimental evaluation

Our test environment is on the 3rd floor of Oakland University library, with seven APs installed. In total 18 survey points are collected for training purposes, as shown in Figure 6. Localization tests were conducted 10 times on a predefined path. The average length of the path is about 85m. During the traverse on the path, we measured the error distance between the estimated location and the ground truth location. The ground truth location is based on manual annotation of waypoints.

Comparison between the estimated path and the ground truth path is shown in Figure 7. The red line represents the ground truth path, while the blue dot and the thick stroke illustrate the estimated path.

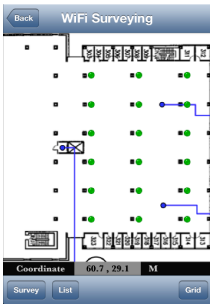

Figure 6: Survey points display on the screen

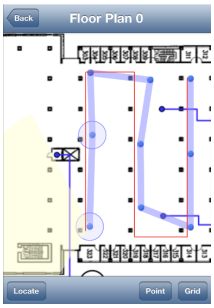

Figure 7: Comparison of the estimated path against the ground truth

The maximum error distance is about 5 m in the corner when we were making a turn. The mean and median error distances are 3.6 m and 2.9 m, respectively. Table 3 shows the advantages of our system when compare to pure W-KNN method. This real time test result clearly demonstrates the high accuracy of our WiFi based indoor localization system.

Error	Mean	Median	Maximum
WiFi iLocate	3.6 m	2.9 m	5 m
W-KNN	6.5 m	6.4 m	>10 m

Table 4: Localization accuracy between WiFi iLocate against weighted KNN

Noted that the above result is achieved with pure WiFi based localization algorithm, we haven't incorporate the information of motion sensors in the iOS device, due to the difficulty in handling the noisy motion data.

Although we have applied Gaussian process to reduce the labor work of collecting WiFi survey point in the offline training phase, this job remains heavy in a large indoor environment. Researchers have tried to overcome this problem using crowdsourcing [13] or simultaneous localization and mapping (SLAM) [14] method. These techniques can be tailored and combined into our system.

V. CONCLUSION AND FUTURE PLAN

In this paper, we have demonstrated an indoor localization system based on Gaussian process modeling of WiFi RSS dataset and particle filter localizer. The simulation result showed that our algorithm yields promising location estimation. The implementation on iOS platform and the test in real world situation proved that WiFi iLocate is a reliable, real-time, press-to-go indoor localization system. To the best of our

knowledge, WiFi iLocate is the first app delivering such accurate, highly integrated indoor localization system on smartphone. Based on our system, many location-aware applications will be able to function properly indoor, provide more convenient service to people's daily life.

In the future, we will combine motion sensors on the smartphone, in hope that multimodality could provide more location-related information and help us develop a sophisticated motion model to improve the localization performance.

REFERENCES

[1] E. Chan and G. Baciu. *Introduction to Wireless Localization*, John Wiley & Sons Singapore Pte. Ltd, 2012.

[2] P. Bahl and V. N. Padmanabhan. "RADAR: An in-building RF-based user location and tracking system" In *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM' 00)*, pages 775-784, March 2000.

[3] M. Youssef. "HORUS: A WLAN-based indoor location determination system" Department of Computer Science, University of Maryland, 2004.

[4] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg. "Compass: A probabilistic indoor positioning system based on 802.11 and digital compasses" In *Proceedings of the 1st international workshop on Wireless network test beds, experimental evaluation & characterization*, pp. 34-40. ACM, 2006.

[5] Shopkick application. http://www.shopkick.com/

[6] H. Liu, Y. Gan, J. Yang, S. Sidhom, Y. Wang, Y. Chen and F. Ye. "Push the Limit of WiFi based Localization for Smartphones" *ACM MobiCom* 2012.

[7] L. Agrawal, D. Toshniwal. "Smart Phone Based Indoor Pedestrian Localization System" *13th International Conference on Computational Science and Its Application*, 2013.

[8] B. Ferris, D. Hahnel, and D. Fox. "Gaussian processes for signal strength-based location estimation" in *Proceedings of Robotics: Science and Systems*, August 2006.

[9] F. Duvallet and A. Tews. "WiFi position estimation in industrial environments using gaussian processes" in *IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Pages 2834-2839, 2009.

[10] A. Schwaighofer, M. Grigoras, V. Tresp, and C. Hoffmann. "GPPS: A Gaussian Process Positioning System for Cellular Networks" in *Advances in Neural Information Processing Systems*, 2003.

[11] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[12] H. Liu, H. Darabi, P. Banerjee, and J. Liu. "Survey of wireless indoor positioning techniques and systems" *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 37(6): 1067–1080, 2007.

[13] A. Rai, K. Chintalapudi, V. Padmanabhan and R. Sen. "Zee: Zero-Effort Crowdsourcing for Indoor Localization" In *Proceedings of the 18th annual international conference on Mobile computing and networking*, August 22-26, 2012, Istanbul, Turkey.

[14] B. Ferris, D. Fox and N. Lawrence. "WiFi-SLAM using Gaussian process latent variable models" In *Proceedings of IJCAI 2007*, pp. 2480-485, 2007.