Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 545792, 13 pages
http://dx.doi.org/10.1155/2015/545792

Research Article

Real-Time and Accurate Indoor Localization with Fusion Model of Wi-Fi Fingerprint and Motion Particle Filter

Xinlong Jiang,[1,2,3] Yiqiang Chen,[1,2] Junfa Liu,[1,2] Dingjun Liu,[4] Yang Gu,[1,2,3] and Zhenyu Chen[1,2,3]

[1]*Beijing Key Laboratory of Mobile Computing and Pervasive Device, Beijing 100190, China*
[2]*Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*
[3]*University of Chinese Academy of Sciences, Beijing 100190, China*
[4]*Xiangtan University, Hunan 411105, China*

Correspondence should be addressed to Yiqiang Chen; yqchen@ict.ac.cn

Received 24 August 2014; Revised 4 November 2014; Accepted 31 December 2014

Academic Editor: Tao Chen

As the development of Indoor Location Based Service (Indoor LBS), a timely localization and smooth tracking with high accuracy are desperately needed. Unfortunately, any single method cannot meet the requirement of both high accuracy and real-time ability at the same time. In this paper, we propose a fusion location framework with Particle Filter using Wi-Fi signals and motion sensors. In this framework, we use Extreme Learning Machine (ELM) regression algorithm to predict position based on motion sensors and use Wi-Fi fingerprint location result to solve the error accumulation of motion sensors based location occasionally with Particle Filter. The experiments show that the trajectory is smoother as the real one than the traditional Wi-Fi fingerprint method.

1. Introduction

Nowadays, indoor location and tracking are very important in our daily life. Indeed, many applications require timely and accurate location information of the mobiles (context-aware application, emergency situation...) [1, 2]. But Global Positioning System (GPS), Assisted GPS (AGPS), and Mobile Base Station (MBS) cannot be used in indoor areas, as the signals cannot reach indoor area.

Fortunately, now many buildings are equipped with WLAN Access Points (APs), such as office buildings, hospitals, shopping malls, museums, and airports. It becomes easy to offer indoor location services without any other infrastructure investment. A common idea of Wi-Fi based indoor location is fingerprint method [3–8]. It is a classical classification problem where different supervised machine learning techniques have been used. A lot of researches show that Wi-Fi signal can generate good prediction for indoor localization [9], but it will spend some time to scan the Wi-Fi signal around so that it cannot offer continuous location service.

Nowadays, the smartphone is the most wildly used device, in part due to its abundant inertial sensors, such as accelerometer, magnetometer and gyroscope. They can be used to measure people's motion state, including walking speed and orientation. Based on this information, we can estimate the relative changing of position by determining a pedestrian's movement. This kind of methods is based on the Inertial Navigation System (INS) [10] method, which can offer continuous localization. But due to its iterative process, we can only estimate the position with high accuracy in a short time. The error accumulation will lead the gap between the movement and the estimated result increases continuously.

By the way, the power consumption is also important. As far as we know, the battery usage of Wi-Fi and communication module is several dozen times higher than the Inertial Measurement Unit (IMU) that consists of gyroscopes and accelerometers. Carroll and Heiser [11] also did some analysis of power consumption in smartphones and the same conclusions can be seen.

Therefore, in this paper, we combine the advantages of two methods and present real-time and accurate indoor localization based on fusion model of Wi-Fi fingerprint and motion Particle Filter. We take use of the accurate Wi-Fi location result to modify the motion Particle Filter in the correction step, so to eliminate the error accumulation caused by motion sensors based location. And this system includes three main modules: sensors data based location model (ELM regression); Wi-Fi based location model (ELM + K-NN classification); and the fusion Particle Filter model.

The rest of the paper is organized as follows. We firstly review some related works of indoor location and tracking in Section 2. Then we give a brief introduction of proposed method in Section 3. Section 4 demonstrates the feature selection for sensors data. In Section 5, we introduce ELM algorithm. After that, we introduce Particle Filter for sensors based location in Section 6. In Section 7, we introduce the fusion model of Wi-Fi fingerprint and motion Particle Filter. Experiments and performance evaluation are given in Section 8. In the end, we make a short conclusion in Section 9.

2. Related Works

As the basic support of Indoor LBS application, a timely localization with high-accuracy and low power consumption is very important. At present, one of the most popular techniques is Wi-Fi fingerprint based location [3–8]. References [3–8] take use of the existing wireless network infrastructures to avoid extra deployment costs.

Wi-Fi fingerprint based locations contain two phrases: offline training and online mapping. During the offline phase, a fingerprint information map will be built. In this map, there are some registered points and corresponding Received Signal Strength (RSS) from the APs around. On the online phase, people receive the Wi-Fi signals, and then some machine learning algorithms will be used to predict the position based on fingerprint dataset. It becomes a classical classification problem where different supervised machine learning techniques have been used to train classifiers, using the signal strength from different APs as the feature and providing the location estimation as their output estimation. Nearest Neighbor (K-NN), Decision Tree (DT), Bayesian, Support Vector Machine (SVM), and ELM [12] are most frequently used by location fingerprint. Among all the algorithms above, ELM is more and more widely used for its competitive fast learning speed during both offline and online phrases.

Although Wi-Fi signals can generate good prediction model for indoor position estimation, it will take a while to scan the Wi-Fi signals so that it cannot offer continuous location service. As shown in Figure 1, we select four different android smartphones (HTC G12, Sony LT26ii, HTC G10, and Xiaomi 1S). With each smartphone, we scan the Wi-Fi for 10 times (Test. 1–10). We can conclude that every time we collect the Wi-Fi signals, it will take more than 1 second on average. That is to say, Wi-Fi based location cannot offer immediately location result due to the time consumption of signals scanning.

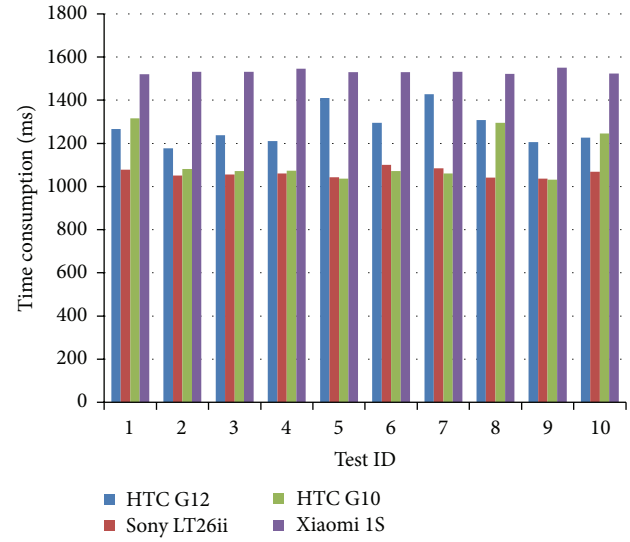

Figure 1: Time consumption of Wi-Fi signal scanning.

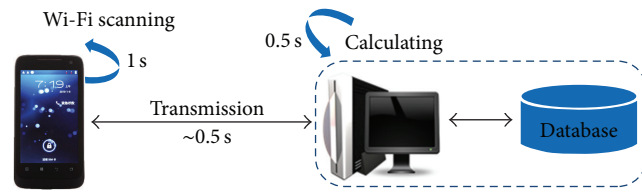

Figure 2: Location system with C-S (Client-Server) architecture.

According to the experiments on Wi-Fi based location system developed by our lab, shown in Figure 2, we can get one location result in every 2 seconds including some other time consumption of transmission and calculation.

In order to overcome the shortcoming of Wi-Fi based location, we can take other measures which can offer continuous location service. Nowadays, as a lot of sensors are integrated in smart phones, so that we can use them to measure people's motion state, including distance and orientation. Distance estimation mainly based on pedestrian model, including step detection and a step model. Step detection typically relies on peak detection over accelerometer [13]. But the result is not good enough, because it is sensitive to noise and other irrelevant motions, producing a high rate of false positives. And there are also some other methods that rely on detecting features such as zero-cross point [14]. Orientation is relatively easier to get via magnetometer, gyroscope, and accelerometer data so that we can get user heading orientation information with high accuracy easily. The orientation of smartphones themselves can be obtained from the motion sensors Application Programming Interface (API) [15] or computed from the raw sensors data.

But based on motion sensors, we can only estimate the position with high accuracy in a short time; it is difficult to locate independently for a long time period because of the error accumulation. To solve this problem, a lot of previous works have been done. Li et al. [16] not only developed an algorithm for reliable detection of steps, heading directions,

and accurate estimation of step length, but also built an end-to-end location system integrating these models. But they still did not solve the problem of location error accumulation. Evennou et al. [1] applied a developed Particle Filter method to WLAN location determination technique and proposed an indoor mobile positioning system. They used the Particle Filter and its constraint on a Voronoi diagram to represent an interesting, so as to handle the variations of the signal strength measurements. It can also aggregate different information like signal strength and the map to obtain correct trajectories without wall-crossings. But in real applications, it is difficult to get the geography Information.

From the related works above, we can see that Wi-Fi based location can offer high accuracy, but it needs high time and power consumptions. Inertial sensors based location estimates the position with high accuracy in a short time, but the shortage is the error accumulation. Thus, in this paper, we present real-time and accurate indoor location based on fusion model of Wi-Fi fingerprint method and motion Particle Filter.

3. Overview of Our Method

The architecture of our method is shown in Figure 3. It contains three main parts. (1) Sensors data based short-time location with ELM regression: we can determine the orientation and velocity to tracking users, but as every location result is based on the former one, it will cause error accumulation. (2) Wi-Fi based location with ELM classification and K-NN: we can get high location accuracy with this method, but as the time consumption of Wi-Fi signals scanning and data transmission, we can only get discrete result in every several seconds. (3) Fusion Particle Filter model: in order to solve the problems in the two methods mentioned above, we add a fusion Particle Filter model to eliminate the error accumulation caused by motion sensors based location with Wi-Fi location result in the correction step and offer a smooth location trajectory.

4. Feature Selection for Sensors Data

Currently, there are a lot of sensors integrated in smartphones. Some previous methods only directly use the API data, such as orientation. But sometimes, the API data cannot offer high precision. As shown in Figure 4, the orientation directly offered by API has a big bias with the real one caused by the environment changing. So in this paper, we use regression learning algorithm with a variety of sensors data to determine the motion state.

In order to find the strong correlation between movement state and sensors data, we investigate the existing sensors on the smartphone [15]. Generally, there are seven sensors that can reflect motion state, shown in Table 1.

When we mention localization, we want to get the orientation and velocity from sensors data. Among all the sensors above, the accelerometer, magnetic field, orientation, gyroscope, and gravity sensors have correlation with the orientation and velocity. We tried different combinations of them to predict the orientation and velocity. Finally, we find that velocity has a strong relationship with accelerometer x, y, and z axes and gyroscope x, y, and z axes, as seen in the following formula:

$$v \sim \{\text{acc}.x, \text{acc}.y, \text{acc}.z, \text{gyr}.x, \text{gyr}.y, \text{gyr}.z\}. \quad (1)$$

And we use orientation x, y, and z axes, magnetic field x, y, and z axes as the features to determine orientation, as seen in the following formula:

$$o \sim \{\text{ori}.x, \text{ori}.y, \text{ori}.z, \text{mag}.x, \text{mag}.y, \text{mag}.z\}. \quad (2)$$

The reason why we choose $\{\text{acc}.x, \text{acc}.y, \text{acc}.z, \text{gyr}.x, \text{gyr}.y, \text{gyr}.z\}$ as the feature to predict v is that different walking speed

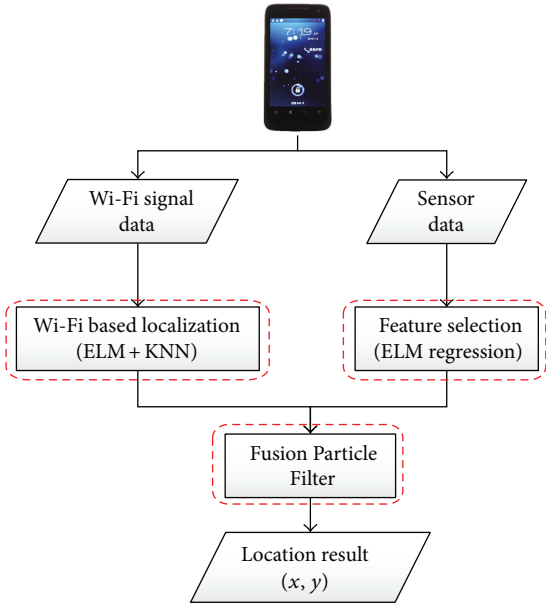

Figure 3: Architecture of our proposed method.

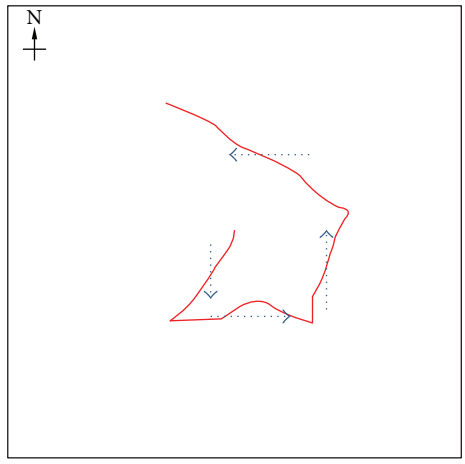

Figure 4: Orientation API output.

Table 1: Motion sensors in smart phone.

ID	Sensor	# descriptions	# units of measure
1	Accelerometer	Acceleration force along the x, y, and z axes	m/s^2
2	Magnetic field	Measures the ambient geomagnetic field for all the three physical axes (x, y, and z)	μT
3	Orientation	Measures degree of rotation that a device makes around all three physical axes (x, y, and z).	$^\circ$
4	Gyroscope	Rate of rotation around the x, z, and z axes.	rad/s
5	Gravity	Force of gravity along the x, y, and z axes.	m/s^2
6	Linear acceleration	Acceleration force along the x, y, and z axes.	m/s^2
7	Rotation vector	Rotation vector component along the x, y, and z axes.	Unitless

causes different shaking degree, which can be observed from the accelerometer readings. As shown in Figure 5, we can conclude that this user walks faster and faster.

We use the same way to choose {ori.x, ori.y, ori.z, mag.x, mag.y, mag.z} as the feature vector to predict orientation o. Although orientation sensor's output cannot match the real orientation very well, it can still offer some positive effect. Magnetic field measures the ambient geomagnetic field which has strong relationship with orientation.

In order to use machine learning algorithm to predict velocity and orientation, we firstly prepare training data.

Firstly, experiment participants walk along a fixed route with a smart phone. Motion sensor data will be collected, and participants will label the time at given points. The frequency of sensors is different in each smartphone. In our experiments $f_s \approx 65$ Hz.

Secondly, we use Cubic Spline Interpolation [17] to smooth the trajectory.

Thirdly, we use slide-window to segment the trajectory with window length T_w. The T_w will be determined in the experiment, but we must keep $T_w > 1/f_s$ so that in every single slide-window there is at least one integrated data. Let us take x component of acceleration in ith window as an example; we use the average of the sensors data as the feature according to

$$\text{acc}_i.x = \frac{1}{N_i}\sum_{k=1}^{N_i}\text{acc}_{ik}.x, \tag{3}$$

where N_i is the data number collected by the smartphone.

According to the smooth trajectory, in the ith stage, we can measure the orientation o_i and velocity v_i, which can be treated as the label for regression algorithm.

5. ELM Algorithm

After feature selection, we are going to predict the motion state. Commonly, the accelerometer has been used to count the number of steps. Then estimate the distance with the user's steps stride [18]. But it depends on the assumption that user's steps strides are always the same, which is hard to satisfy in practical applications.

To avoid this assumption, we adopt ELM to train an effective model which can offer real-time prediction of orientation and walking speed.

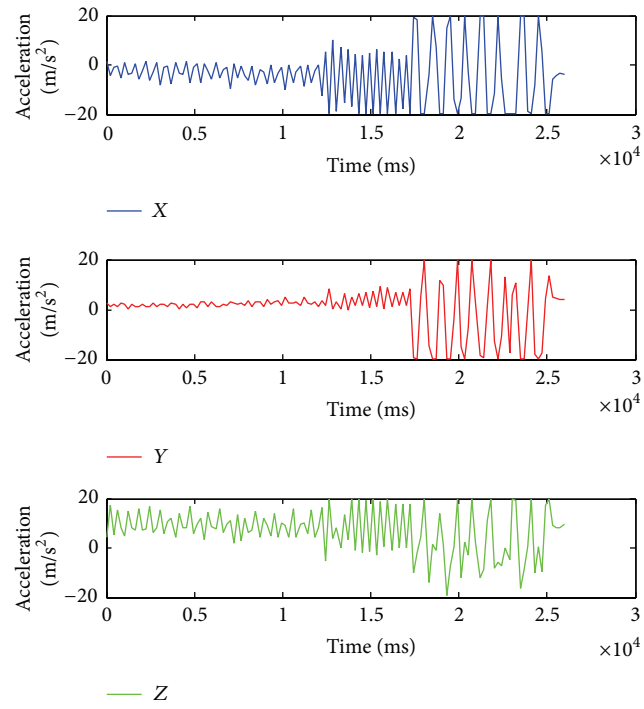

Figure 5: Accelerometer and walking speed.

ELM [12] is an Artificial Neutral Network (ANN), especially Single Layer Feedforward Networks (SLFN), developed by Huang et al. Given N arbitrary distinct samples $(\mathbf{x}_i, \mathbf{t}_i) \in R^n \times R^m$, $i = 1, 2, \ldots, N$. Here, $\mathbf{x}_i$ is an $n \times 1$ input vector $\mathbf{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}]^T$ and $\mathbf{t}_i$ is an $m \times 1$ target vector $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{im}]^T$. The network with L hidden nodes is shown in Figure 6. The output function of this network can be represented as follows:

$$f_L\left(\mathbf{x}_j\right) = \sum_{i=1}^{L}\beta_i G\left(\mathbf{a}_i, b_i, \mathbf{x}_j\right), \quad j = 1, \ldots, N, \tag{4}$$

where $\mathbf{a}_i$ and b_i are the learning parameters of hidden nodes and β_i is the weight connecting the ith hidden node to the output node. $G(\mathbf{a}_i, b_i, \mathbf{x})$ is the output of the ith hidden node with respect to the input $\mathbf{x}$. For additive hidden node with

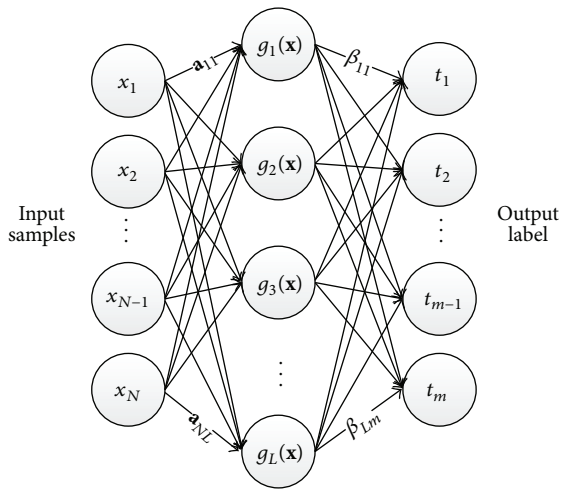

FIGURE 6: SLFN with L hidden neurons.

the activation function $g(x) : R \rightarrow R$ (e.g., sigmoid and threshold), $G(\mathbf{a}_i, b_i, \mathbf{x})$ is given blow:

$$G(\mathbf{a}_i, b_i, \mathbf{x}) = g(\mathbf{a}_i \cdot \mathbf{x} + b_i), \quad b_i \in R. \tag{5}$$

If an SLFN with L hidden nodes can approximate these N samples with zero errors, this then implies that there exist β_i, $\mathbf{a}_i$, and b_i such that

$$f_L(\mathbf{x}_j) = \sum_{i=1}^{L} \beta_i G(\mathbf{a}_i, b_i, \mathbf{x}_j) = \mathbf{t}_j, \quad j = 1, \ldots, N. \tag{6}$$

Equation (6) can be summarized as

$$H\beta = T, \tag{7}$$

where

$$H(\mathbf{a}_1, \ldots, \mathbf{a}_L, b_1, \ldots, b_L, \mathbf{x}_1, \ldots, \mathbf{x}_L) = \begin{bmatrix} G(\mathbf{a}_1, b_1, \mathbf{x}_1) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ G(\mathbf{a}_1, b_1, \mathbf{x}_N) & \cdots & G(\mathbf{a}_L, b_L, \mathbf{x}_N) \end{bmatrix}, \tag{8}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m}, \qquad T = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_N^T \end{bmatrix}_{N \times m}. \tag{9}$$

According to [12], the hidden node parameters $\mathbf{a}_i$ and b_i (input weights and biases or centers and impact factors) of SLFNs need not be tuned during training and may simply be assigned with random values. Equation (7) then becomes a linear system and the output weights β are estimated as

$$\hat{\beta} = H^{\dagger}T = \left(H^T H\right)^{-1} H^T T, \tag{10}$$

where $H^{\dagger}$ is the Moore-Penrose generalizes inverse of hidden layer output matrix H. Equation (10) can be seen as a linear system; the least-squares solution of (8) is the answer of ELM.

By now, ELM and some other related algorithms such as OS-ELM [19, 20], SELM [21, 22], and Unsurprised-ELM [22] have been developed and wildly applied in some real applications. Huang [23] made a further insight into ELM, including some discussion about random neurons, random features, and kernels. And in [24, 25], Cao et al. made some improvement to the basic ELM and achieved a good performance. Because of the fast learning speed and good learning ability, ELM has been wildly used in ubiquitous computing applications, such as activity recognition [26–29]. That is why we use ELM model both in offline learning and in online prediction. Our lab has done a lot of Wi-Fi indoor location research based on it and achieved good performance [21, 30].

From all the information above, we can see that ELM model can be used in regression and classification problems. In our application, the velocity and orientation are all continuous variable. So we use ELM as a regression model.

(i) ELM Model for Velocity. For velocity, the input vector is a vector with six features:

$$\mathbf{x}_i = \left[\text{acc}_i.x, \text{acc}_i.y, \text{acc}_i.z, \text{gry}_i.x, \text{gry}_i.y, \text{gry}_i.z\right]^T \tag{11}$$

and the output vector is only a scalar value $\mathbf{t}_i = [v_i]^T$.

(ii) ELM Model for Orientation. For orientation, the input is a vector with four features:

$$\mathbf{x}_i = \left[\text{ori}_i.x, \text{ori}_i.y, \text{ori}_i.z, \text{mag}_i.x, \text{mag}_i.y, \text{mag}_i.z\right]^T \tag{12}$$

and the output vector in also a scalar value $\mathbf{t}_i = [o_i]^T$.

According to [31], we also have offline and online phases. During offline phases, we will collect the sensors data and the trajectory to train these two models mentioned above. During the online phases, we will use sensors data and models to determine v and o. Then we can use (13) to calculate the current position. Consider

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{cases} \begin{bmatrix} x_{t-1} \\ y_{t-1} \end{bmatrix} + \begin{bmatrix} T_w & 0 \\ 0 & T_w \end{bmatrix} * \begin{bmatrix} v_t & 0 \\ 0 & v_t \end{bmatrix} * \begin{bmatrix} \sin(o_i) \\ \cos(o_i) \end{bmatrix}, & t > 1, \\ \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}, & t = 0. \end{cases} \tag{13}$$

As our method is based on an iterative process, so the initial position $[x_0 \quad y_0]^T$ should be given.

By now, we can already track the trace, given the initial position, in every T_w second; we collect the sensors data and

use these two ELM models to determine v and o. So that, the current position will be calculate by (13). But unfortunately, sensors on smart phone are not accurate enough to offer data with high confidence. For a short time, it is really a reliable method, but as time goes on, error accumulation will destroy the reliability.

6. Particle Filter for Sensors Based Location

Particle Filter [32] is an on-line posterior density estimation algorithm estimating the posterior density of the state-space by directly implementing the Bayesian recursion equations. It provides a well-established methodology for generating samples from the required distribution without requiring assumptions about the state-space model or the state distributions. The state-space model can be nonlinear and the initial state and noise distributions can take any form required. For a lot of position and navigation problems, common motion model based filters can be applied. Models that are linear in the state dynamics and nonlinear in the measurements are considered [33].

State Transition Equation. Consider

$$\mathbf{x}_{t+1} = \mathbf{A}\mathbf{x}_t + \mathbf{B}_u\mathbf{u}_t + w_t. \tag{14}$$

Observation Equation. Consider

$$\mathbf{y}_t = h\left(\mathbf{x}_t\right) + e_t, \tag{15}$$

where $\mathbf{x}_t$: state vector, $\mathbf{u}_t$: measured inputs, w_t: process noise, $\mathbf{y}_t$: measurements, and e_t: measurement error.

In our application, we turn the model, (14) and (15), into a specific one, as shown in

$$\begin{bmatrix} x_t \\ y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{t-1} \\ y_{t-1} \end{bmatrix} + \begin{bmatrix} T_w & 0 \\ 0 & T_w \end{bmatrix} \begin{bmatrix} v_{xt} \\ v_{y_t} \end{bmatrix} + Q, \tag{16}$$

$$\begin{bmatrix} X_t \\ Y_t \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_t \\ y_y \end{bmatrix} + R. \tag{17}$$

In (16), $[x_t, y_t]^T$ denotes the state vector associated to each particle (position). T_w is the elapsed time between the $(t-1)$th and the tth measurements. Q is the process noise. Here the used process is a zero mean Gaussian noise with a $0.1\,\mathrm{m/s}^2$ variance which is a realistic model of pedestrian movement.

Equation (17) is observation equation, which denotes the relationship between the hidden states $[x_t, y_t]^T$ and the observation states $[X_t, Y_t]$. Because we directly observe the coordinate point from ELM regression algorithm, there is no measurement error in this process, so we set R to be 0.

A Particle Filter approximates the position at time t by a set of N particles. Each particle contains a position z_k^i and a weight $P_r[x_t \mid z_t^i]$. $P_r[x_t \mid z_t^i]$ indicates the importance of the ith particle. In the case of an indoor movement, the following law has been retained:

$$P_r\left[x_t \mid z_t^i\right] = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{\left(X_{xt} - X_{z_t^i}\right)^2}{2 \cdot \sigma^2}\right]. \tag{18}$$

With X_{x_t} the position returned by the ELM regression algorithm, $X_{z_t^i}$ the position of the ith particle at time t, and σ the measurement confidence. The smaller σ will be, the more confident the user is in the measurement. That would mean that there is very little variation in the measurements for the same position. Here, $\sigma = 50$ was chosen.

According to the process of Particle Filter algorithm, there are four steps in a circle: prediction, correction, particle update, and resampling. At each T_w second, we will determine a new position by the following steps.

(i) Feature Selection

(1) We collect sensors data from mobile and calculate the feature vector according to (3).

(2) Then we use two ELM regression models to calculate the orientation o_i and velocity v_i.

(ii) Then We Go to Particle Filter Phase

(3) Prediction: during this step, every particle will propagate to a new one based on state transition equation (16). After that, all the particles move to new places.

(4) Correction: now, we have all the particles and we must give them the weight; according to o_i, v_i, and location of last moment, we can calculate the observed value by (17). Then we can use (18), from which we can see that the more particle far away from the observed value, the less value the particle has.

(5) Particle Update: the weight update equation is given in [33, 34]:

$$w_t^i = w_{t-1}^i \cdot P_r\left[x_t \mid z_t^i\right]. \tag{19}$$

To obtain the posterior density function, we have to normalize those weights. After a few iterations, only a few particles can still alive. So we should go to resampling step to avoid having few remaining particles.

(6) Resampling: the resampling step is the critical point for the PF. The fundamental idea is to remove the particles which have too low weight. Various resampling algorithms can be chosen. We choose the Sequential Importance Resampling (SIR) for the reason that it is a simple one which is wildly used in SLAM (Simultaneous Location and Mapping) [35]. Take N samples with replacement from the set $\{x_t^i\}_{i=1}^N$, where the probability to take ith sample is w_t^i. Let $w_t^i = 1/N$.

(7) Finally, we can calculate the new location result by $\widehat{x}_t = \sum_{i=1}^N w_t^i x_t^i$. And then go to step (1) until stop.

7. Fusion Model of Wi-Fi Fingerprint and Motion Particle Filter

By now, only the sensors data are used for localization. But they cannot offer long time location service with high accuracy due to the error accumulation. To solve this problem,

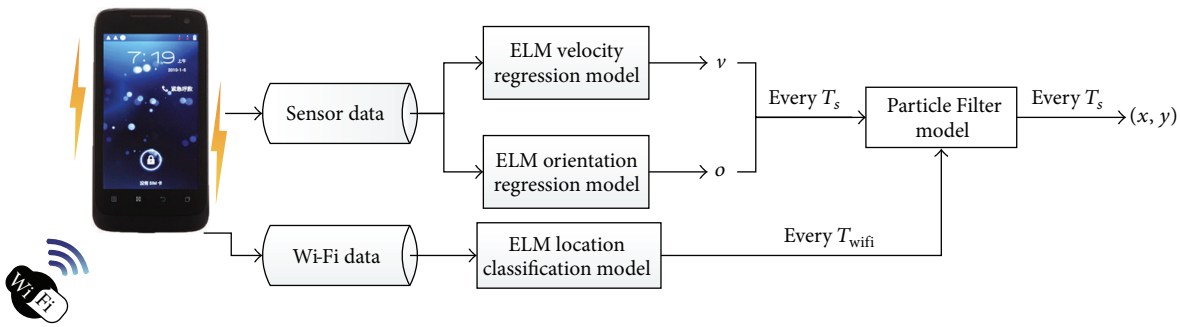

Figure 7: Location process of the whole system.

we present a fusion Particle Filter model, combining the advantages of motion sensors based location and Wi-Fi based location.

Wi-Fi fingerprint based location methods include offline learning and online prediction. During the offline phrase, we collect fingerprints, including coordinate $\{x, y\}$ and Wi-Fi Received Signal Strength Indication (RSSI) $\{r_1, r_2, \ldots, r_k\}$ measured at $\{x, y\}$. Actually, we do not know the total number of Wi-Fi Aps and their position. We firstly calibrate all the location area and then choose the several most frequent APs as the feature. After that, we change all the fingerprints to only contain these frequent APs and supplement the missing items with default value −95 dB.

According to [36], during the online phase, we use ELM classification and K-NN algorithm.

With ELM, we can find the K Nearest Neighbor points, but we do not know the distance because the ELM only offers the nearest points but not the weights. So we use the Euclidean distance of signal strength to determine the weight according to

$$\begin{aligned} \mathrm{dis}_k &= \sqrt{\sum_{i=1}^{n} \left(\mathrm{rss}_{ki} - \mathrm{rss}_i'\right)^2}, \\ w_k &= \frac{\sum_{i=1}^{K} \mathrm{dis}_i - \mathrm{dis}_k}{(n-1)\sum_{i=1}^{K} \mathrm{dis}_i}. \end{aligned} \tag{20}$$

When we use smart phone to do Wi-Fi based indoor location, it takes a little while to collect Wi-Fi data. So we can only get a location result in every $T_{\text{Wi-Fi}}$ second. To eliminate the error accumulation caused by sensors based location, we add the Wi-Fi location result in particle update step and resampling step of Particle Filter.

As shown in Figure 7, once the smartphone begins to locate, the Wi-Fi signals and motion sensors data will be collected. By using the motion sensors data, we get a location result in every 200 milliseconds. And Wi-Fi based location data will arrive in about every 2 seconds.

We promote the Particle Filter mentioned in last section to a fusion one by adding the Wi-Fi location result in the correction step. When we go to the correction step, we should check whether there comes a Wi-Fi location result. If yes, we use the Wi-Fi location point as the observation value to calculate all particles' weight. And if not, we still use the result predicted by motion sensors data. With the fusion model, our system can offer a highly accurate location result with limited time.

Table 2: Mobile phones' hardware parameters.

	# OS	# CPU (Hz)	# RAM	# ROM
HTC Desire S	Android, v2.3	1 G	768 MB	1 GB
Samsung Galaxy S3	Android, v4.0	1.4 G	1 GB	16 GB
HTC Desire S	Android, v2.2	1 G	768 MB	1.5 GB
SONY Lt26ii	Android, v4.0	1741 M	1 GB	32 GM
HUAWEI S8600	Android, v2.3.4	624 M	512 MB	512 MB

8. Experiments and Performance Evaluation

In this section, we do several experiments to evaluate the performance of our method. All the experiments are running on smartphones and a computer with following configuration.

Smart Mobile Phones. See Table 2.

Computer

Operation System: Windows XP Professional SP3.

CPU: Intel Pentium(R) 4 CPU.

Main frequency: 3.2 GHz.

RAM: 2 G.

8.1. Experimental Design and Data Preparing. In order to evaluate the performance, we build a test platform in our work space, as shown in Figure 8. Including the aisle, it is about 100 m^2 (8.5 m ∗ 12 m).

For the reason that we apply ELM algorithm in both Wi-Fi based location and senor based location, we will collect data to train the model during offline phase.

Concerning sensors data, we plan a fixed route including straights and curves, shown in Figure 9. People walk along this trajectory with any speed: slow, middle, fast, and even run. And they record the time at every label point by touching the volume key. As shown in Figure 10. In order that we can measure different types of motion, there are four persons doing the collection work, and each person walks five times. Every smart phone will be used once by each person.

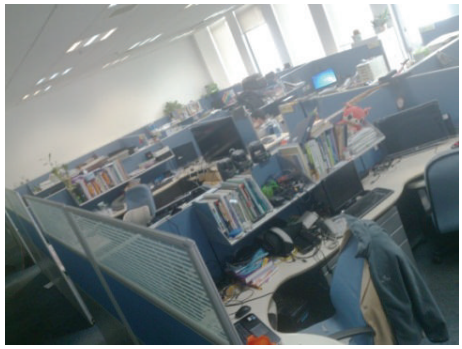

Figure 8: Our working space in ICT F8.

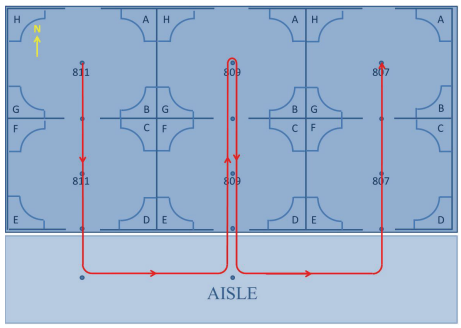

Figure 9: The presetting route.

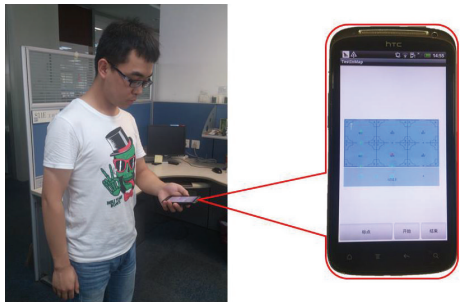

Figure 10: Collecting the sensors data.

After collecting the original sensors data, we use Cubic Spline Interpolation to smooth the trajectory. Thirdly, we use slide-window to segment the trajectory with window length T_w, shown in Figure 11. The T_w will be determined in the experiment, but we must make sure $T_w > 1/f_s$ so that in every single slide-window there is independent data. In every window, we will calculate feature by (1). And measure the orientation and speed from the red points.

For the Wi-Fi based location part, we use the fingerprint method. So we also have to build a signal distribution map. We select 36 points that can cover the test area, shown in Figure 12. Between every two adjacent points, it is about 1.5 m~2 m. We do not strictly require label points equally distributed in this area. We register four times at each point, collect Wi-Fi data for three times, and then use the mean value.

Table 3: MSE of orientation and velocity in different window length.

	Window length				
	200 ms	400 ms	600 ms	800 ms	1000 ms
MSE of orientation (degree)	25.0529	27.8570	26.8727	26.5759	26.4503
MSE of velocity (m/s^2)	0.3001	0.4020	0.4158	0.4642	0.4435

During the online phase, people were walking along the trajectory, shown in Figure 9, to collect testing data. We let a user to carry a smart phone to collect sensors data and get the Wi-Fi location result from the server. Then we will use these raws to predict the trajectory. After that, we can evaluate the performance by comparing with groundtruth.

8.2. Parameters Selection

8.2.1. Determine the Time Window Length T_w. Firstly, we will determine the window length T_w. For the reason that the sensors frequency is $f_s \approx 65$ Hz. In order to void the influence caused by noise, we use the mean value as the features. So we should ensure that there are about ten samples in each window. That is to say the T_w should be bigger than 160 milliseconds.

We use Mean Square Error (MSE) [37] to measure the accuracy in ELM regression algorithms. The results are shown in Table 3. We can see that we can get lowest MSE in both orientation and velocity. So we set T_w to be 200 ms.

8.2.2. Parameters of ELM Regression Models. For basic ELM, we only need to determine the number of hidden nodes L. The input weights and biases of hidden nodes can be generated randomly. We have collected 5316 training samples and divided them into 10 parts. Then we use 10-fold cross validation to select best parameter; the results are shown in Figure 13.

From Figure 13, we can come to conclusions that, for the ELM regression model of velocity, 500 hidden nodes are suitable and, for orientation regression, we can minimize test error when the model has 1000 hidden nodes. Finally, we set L to be 500 for velocity and 1000 for orientation. And we chose *Sigmoid* [38] function for orientation and *Hardlim* [39] function for velocity.

8.3. Experimental Performance. In this section, in order to validate the proposed method, the comparisons to the existing methods are necessary. As the proposed method contains Wi-Fi and sensors information and takes use of ELM and Particle Filter algorithms, comparison experiments with some related methods should be involved. We firstly consider the Wi-Fi based location only and tried four different methods (K-NN, ELM, ELM + Kalman filter, and ELM + Particle Filter). Then we measure the location error of some sensors based methods and proposed method. Also the error accumulation tests have been considered as it is the key factor of location accuracy. Finally, we give a statistical result of

Table 4: Comparison of the performance of the different methods.

	Sensors data (gait + Particle Filter) [40]	Sensors data (ELM)	Sensors data (ELM + Kalman filter)	Sensors data (ELM + Particle Filter)	Sensors data + Wi-Fi (ELM + Particle Filter)
Mean error (m)	1.542	0.609	0.829	0.633	0.150

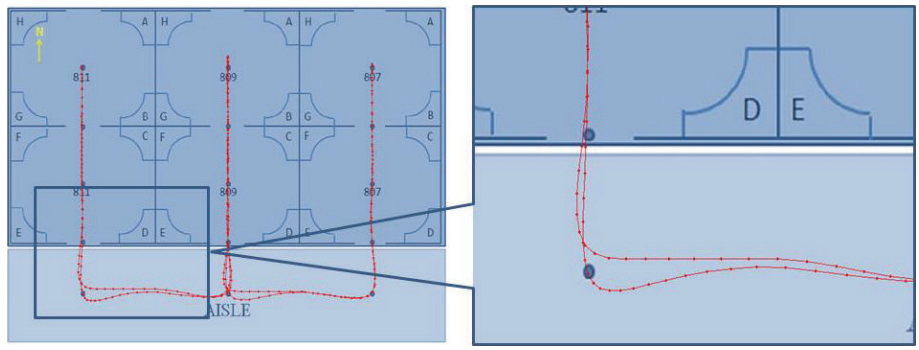

Figure 11: The trajectory after smooth and segmented by slide-window.

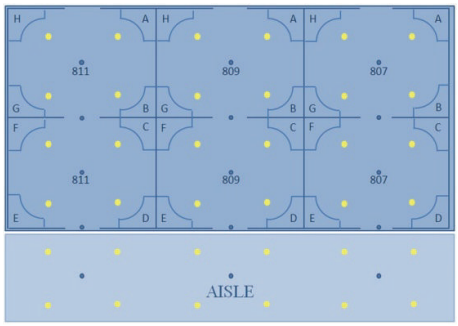

Figure 12: Fingerprint points are in yellow color.

the time consumption in our experiment to state that our proposed method is real practical.

8.3.1. Evaluate the Location Accuracy. Firstly, we compare the location performance of K-NN, ELM, Kalman filter, and Particle Filter based on Wi-Fi fingerprint. As shown in Figure 14, ELM performs better than K-NN, but it is still not very good as expected. We can explain the reason as that every location process is separated with each other. They are not treated as a sequence; thus some relevant information has been missed. So we apply Kalman filter and Particle Filter to the location results of ELM. Fortunately, both of them can raise the accuracy by big percentages. But, Particle Filter performs much better as it is a nonlinear system, which is more effective to simulate the walking pattern.

Besides the Wi-Fi information, sensors data is also very useful for indoor location. It is usually used to detect gait and then use a default step length to estimate the walking distance. In proposed method, we use ELM model to build a mapping relationship of sensors data and walking pattern. Moreover, Kalman filter and Particle Filter also can be used to smooth the initial location results. To evaluate their performance, we not only compare the location mean error, but also calculate the location trajectory using Matlab. From Table 4, we can conclude that, comparing to gait detection based method mentioned in [40], ELM based methods can get better results. And with the Particle Filter, mean error becomes smaller. But we can see that when we apply the Kalman filter to the ELM location result, the mean error is bigger than before. We explain it as the function of Kalman filter is to make the trajectory smoother, not to minimize the mean error, which can be seen in Figure 14. But filters can only make the trajectory smoother; they cannot solve the error accumulation problem. In order to solve the error accumulation, we combine the Wi-Fi location with the sensors data based location to come into being proposed method, which indeed can offer a small mean error.

From Figure 15(a), we can see that trajectory of sensors based location is not very good. As time goes on, the trajectory is getting more and more far away from groundtruth. The reason is that, within a short period of time, the sensors can offer good valuation of walking orientation and speed, but the bias also exists. The next result is based on the previous step, so the bias will cumulate to unbearable error after a period of time. Figure 15(b) shows the trajectory of sensors based location with ELM and Kalman filter. We can see it making the trajectory smoother in some turnings. But it is a little more far from the groundtruth. We explain it as that the function of Kalman filter is to smooth the trajectory, but not to minimize the location error. Figure 15(c) indicates the sensors based location with ELM and Particle Filter. From it, we can conclude that the Particle Filter can make the trajectory smoother than before, but it does not improve the error accumulation by a large scale. The proposed method can

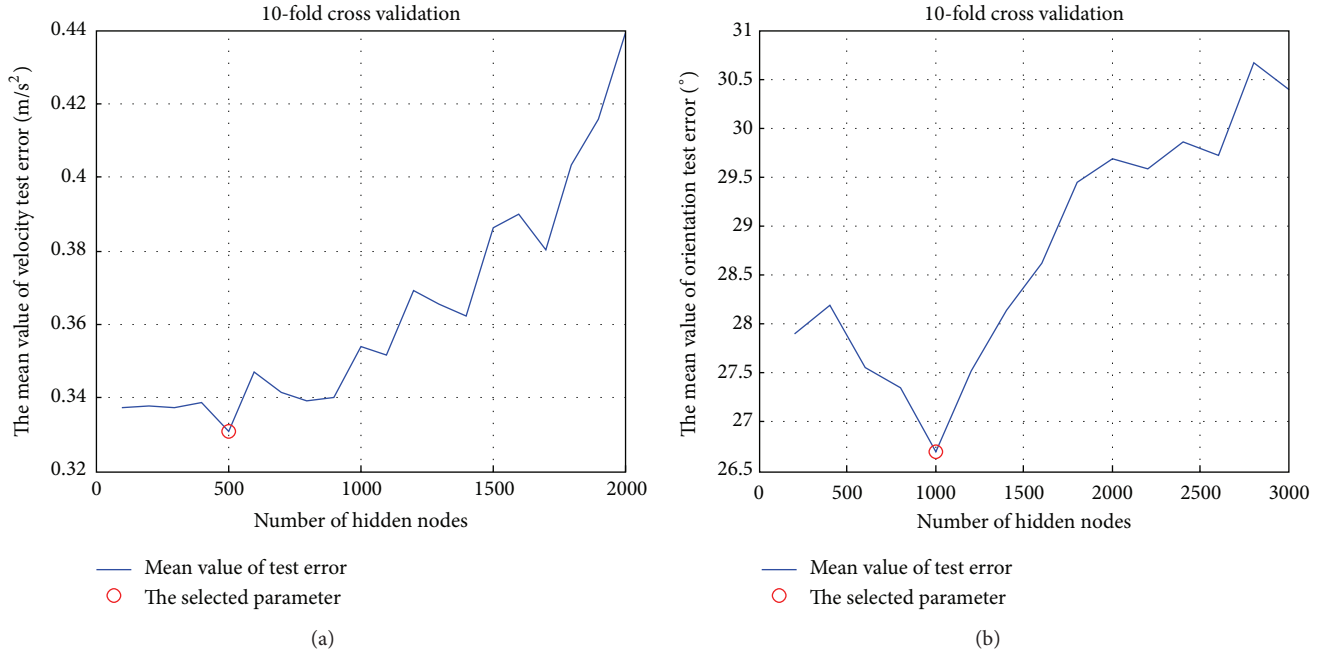

FIGURE 13: 10-fold cross validation to select number of hidden nodes (a) for velocity and (b) for orientation.

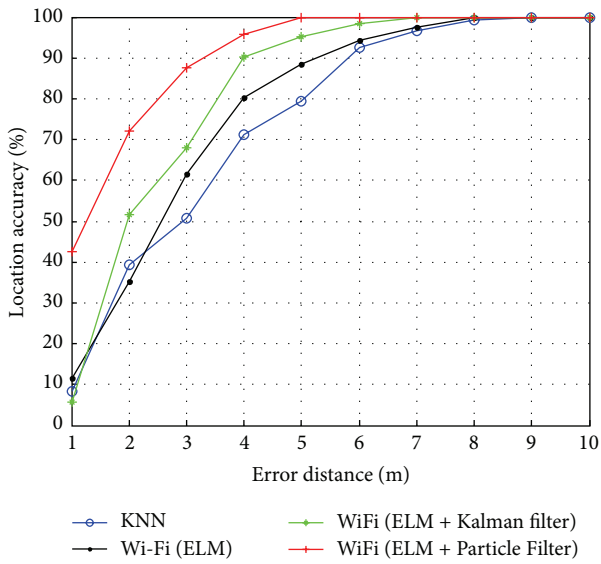

FIGURE 14: Location accuracy of four methods.

be seen in Figure 15(d). It is really better than those methods mentioned above. It can offer a timely and highly accurate location result.

8.3.2. Experiments on Error Accumulation. In order to value the error accumulation existing in sensors based location and how our method improved the accuracy with the help of Wi-Fi based location and positive impact of Particle Filter, we calculate the distance error between the predicted position and the real one. And the result is shown in Figure 16. From that, we can conclude that when we use sensors only, the location error will accumulate into unbearable distance as time goes on. When we added Kalman filter to it, the location error increases more than before, which can be explained as that Kalman filter's function is to smooth the trajectory, not to minimize the location error. Fortunately, when we add Particle Filter, the location error decreases, but the error accumulation still leads the location accuracy divergence. When we use the Wi-Fi based location result to calibrate it within the Particle Filter framework, it will eliminate the error accumulation and keep the predicted position high precision.

8.3.3. Evaluate the Location Time Consumption. We also did the experiment of the time consumption of Wi-Fi based location and our methods.

From Figure 17 we can see that when we use Wi-Fi based location method on each kind of smart phone, it will take about 4.5 seconds. We can see the point indicating the location result jumping on the map. But using our method, the time consuming is almost 200 milliseconds, so that we can get a smoothly trajectory on the map.

9. Conclusion and Future Work

In this paper, we have presented a real-time and accurate indoor localization based on fusion model of Wi-Fi fingerprint and motion Particle Filter. The reason we raised this problem is that: a lot of sensors are integrated in smart phones, so we can easily get sensors data and Wi-Fi signals

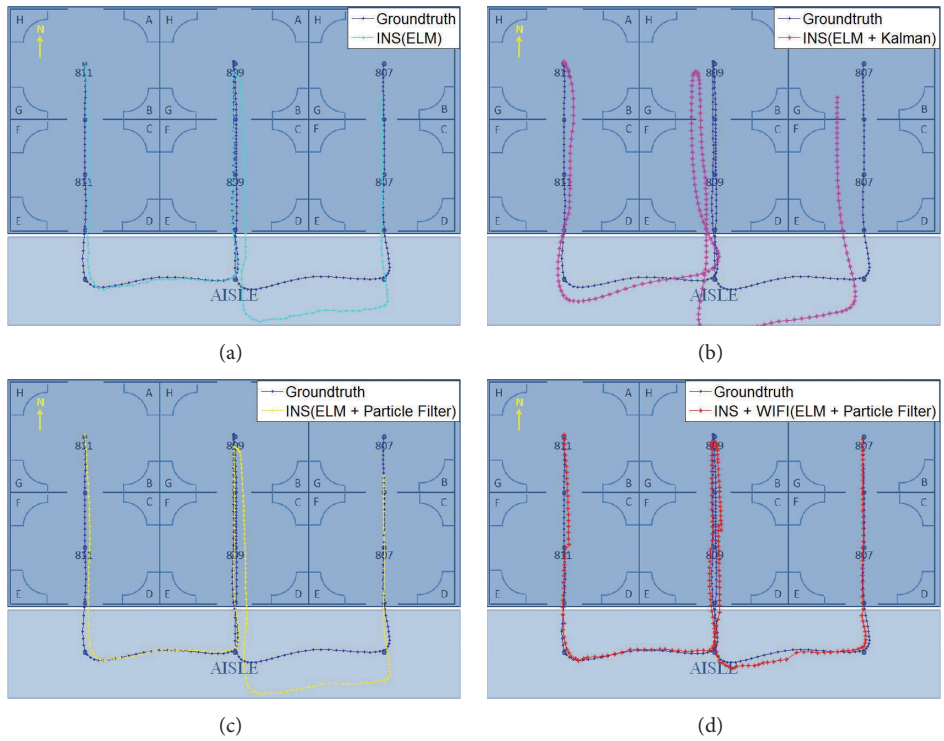

Figure 15: (a) Trajectory of sensors based location with ELM; (b) trajectory of sensors based location with ELM and Kalman filter; (c) trajectory of sensors based location with ELM and Particle Filter; (d) trajectory of our proposed method (INS: sensors data).

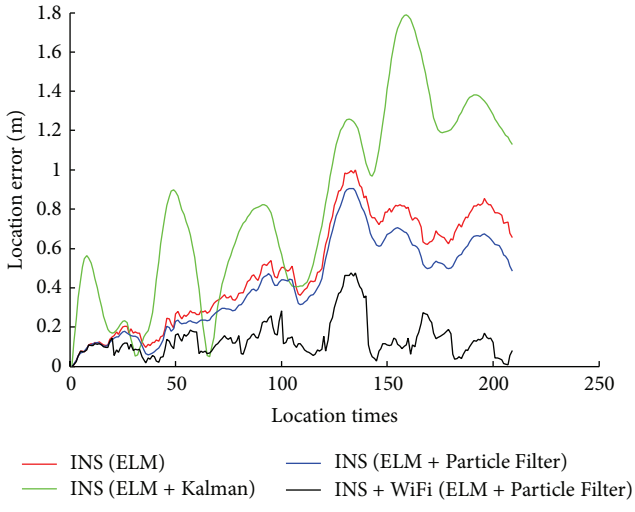

Figure 16: Location error of sensors based service and our proposed method (INS: sensors data).

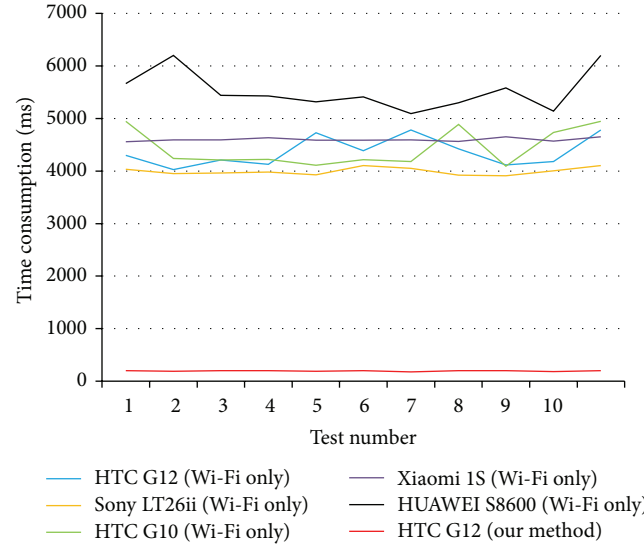

Figure 17: Time consumption of Wi-Fi based location and our method.

which can be used for indoor localization. But any single method cannot get satisfactory result. So we combine Wi-Fi based location and sensors based inertial navigation. And we also use Particle Filter and ELM algorithm to improve the location accuracy. From the experiments result, we can see that our method has achieved a good performance. But to get a better accuracy, we think that another technology should be taken into account. RFID may be the suitable candidate. On the other hand, in order to speed up the localization service, maybe MapReduce framework [41] or Parallel Framework [42–46] can be involved.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by NSFC General Program under Grant nos. 61173066 and 41201410 and Guangdong Province Funds Supported Project for the Development of Strategic Emerging Industry no. 2011912030.

References

[1] F. Evennou, F. Marx, and E. Novakov, "Map-aided indoor mobile positioning system using particle filter," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '05)*, vol. 4, pp. 2490–2494, March 2005.

[2] Z. Chen, "Mining individual behavior pattern based on significant locations and spatial trajectories," in *Proceedings of the IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops '12)*, pp. 540–541, IEEE, March 2012.

[3] P. Bahl and V. N. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '00)*, pp. 775–784, IEEE, March 2000.

[4] A. K. M. M. Hossain, H. Nguyen van, J. Yunye, and S. Wee-Seng, "Indoor localization using multiple wireless technologies," in *Proceedings of the IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS '07)*, pp. 1–8, Pisa, Italy, October 2007.

[5] N. Swangmuang and P. Krishnamurthy, "An effective location fingerprint model for wireless indoor localization," *Pervasive and Mobile Computing*, vol. 4, no. 6, pp. 836–850, 2008.

[6] Z. Sun, Y. Chen, J. Qi, and J. Liu, "Adaptive localization through transfer learning in indoor Wi-Fi environment," in *Proceedings of the 7th International Conference on Machine Learning and Applications (ICMLA '08)*, pp. 331–336, IEEE Computer Society, San Diego, Calif, USA, December 2008.

[7] Y. Chen, J. Qi, Z. Sun, and Q. Ning, "Mining user goals for indoor location-based services with low energy and high QoS," *Computational Intelligence*, vol. 26, no. 3, pp. 318–336, 2010.

[8] Z. Chen, S. Wang, Y. Chen, Z. Zhao, and M. Lin, "InferLoc: calibration free based location inference for temporal and spatial fine-granularity magnitude," in *Proceedings of the 15th IEEE International Conference on Computational Science and Engineering (CSE '12)*, pp. 453–460, Nicosia, Cyprus, December 2012.

[9] Z. Chen, Y. Chen, S. Wang, and Z. Zhao, "A supervised learning based semantic location extraction method using mobile phone data," in *Proceedings of the IEEE International Conference on Computer Science and Automation Engineering (CSAE '12)*, pp. 548–551, Zhangjiajie, China, May 2012.

[10] P. D. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*, Artech House, 2nd edition, 2013.

[11] A. Carroll and G. Heiser, "An analysis of power consumption in a smartphone," in *Proceedings of the USENIX Conference on USENIX Annual Technical Conference*, 2010.

[12] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: a new learning scheme of feedforward neural networks," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 2, pp. 985–990, IEEE, July 2004.

[13] J. Thomas and R. W. Levi, "Dead reckoning navigational system using accelerometer to measure foot impacts," U.S. Patent No. 5,583,776, 1996.

[14] H. Leppäkoski, J. Käppi, J. Syrjärinne, and J. Takala, "Error analysis of step length estimation in pedestrian dead reckoning," in *Proceedings of the 15th International Technical Meeting of the Satellite Division of the Institute of Navigation (ION GPS '02)*, pp. 1136–1142, Portland, Ore, USA, September 2002.

[15] "Motion Sensors," http://developer.android.com/guide/topics/sensors/sensors_motion.html.

[16] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A Reliable and accurate indoor localization method using phone inertial sensors," in *Proceedings of the 14th International Conference on Ubiquitous Computing*, pp. 421–430, ACM, September 2012.

[17] I. J. Schoenberg, *Cardinal Spline Interpolation*, vol. 12, SIAM, Philadelphia, Pa, USA, 1973.

[18] F. Evennou and F. Marx, "Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning," *EURASIP Journal on Applied Signal Processing*, vol. 2006, article 164, 2006.

[19] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Transactions on Neural Networks*, vol. 17, no. 6, pp. 1411–1423, 2006.

[20] J. W. Cao, T. Chen, and J. Fan, "Fast online learning algorithm for landmark recognition based on BoW framework," in *Proceedings of the 9th IEEE Conference on Industrial Electronics and Applications*, pp. 1163–1168, Hangzhou, China, June 2014.

[21] J. Liu, Y. Chen, M. Liu, and Z. Zhao, "SELM: semi-supervised ELM with application in sparse calibrated location estimation," *Neurocomputing*, vol. 74, no. 16, pp. 2566–2572, 2011.

[22] G. Huang, S. Song, J. N. D. Gupta, and C. Wu, "Semi-supervised and unsupervised extreme learning machines," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2405–2417, 2014.

[23] G.-B. Huang, "An insight into extreme learning machines: random neurons, random features and kernels," *Cognitive Computation*, vol. 6, no. 3, pp. 376–390, 2014.

[24] J. Cao and L. Xiong, "Protein sequence classification with improved extreme learning machine algorithms," *BioMed Research International*, vol. 2014, Article ID 103054, 12 pages, 2014.

[25] J. Cao, Z. Lin, G.-B. Huang, and N. Liu, "Voting based extreme learning machine," *Information Sciences*, vol. 185, pp. 66–77, 2012.

[26] Y. Chen, Z. Zhao, S. Wang, and Z. Chen, "Extreme learning machine-based device displacement free activity recognition model," *Soft Computing*, vol. 16, no. 9, pp. 1617–1625, 2012.

[27] Z. Chen, Y. Chen, L. Hu, S. Wang, and X. Jiang, "Leveraging two-stage weighted ELM for multimodal wearables based fall detection," in *Proceedings of ELM-2014 Volume 2: Applications*, vol. 4 of *Proceedings in Adaptation, Learning and Optimization*, pp. 161–168, Springer International Publishing, Cham, Switzerland, 2015.

[28] Z. Chen, S. Wang, Z. Shen, Y. Chen, and Z. Zhao, "Online sequential ELM based transfer learning for transportation mode recognition," in *Proceedings of the 6th IEEE International Conference on Cybernetics and Intelligent Systems (CIS '13)*, pp. 78–83, IEEE, November 2013.

[29] Z. Zhao, Z. Chen, Y. Chen, S. Wang, and H. Wang, "A class incremental extreme learning machine for activity recognition," *Cognitive Computation*, vol. 6, no. 3, pp. 423–431, 2014.

[30] J.-F. Liu, Y. Gu, Y.-Q. Chen, and Y.-S. Cao, "Incremental localization in WLAN environment with timeliness management," *Chinese Journal of Computers*, vol. 36, no. 7, pp. 1448–1455, 2013.

[31] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.

[32] K. Nummiaro, E. Koller-Meier, and L. van Gool, "An adaptive color-based particle filter," *Image and Vision Computing*, vol. 21, no. 1, pp. 99–110, 2003.

[33] F. Gustafsson, F. Gunnarsson, N. Bergman et al., "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[34] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[35] S. Thrun, "Simultaneous localization and mapping," in *Robotics and Cognitive Approaches to Spatial Mapping*, pp. 13–41, Springer, Berlin, Germany, 2008.

[36] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: wireless indoor localization with little human intervention," in *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (MobiCom '12)*, pp. 269–280, ACM, August 2012.

[37] M. Tüchler, A. C. Singer, and R. Koetter, "Minimum mean squared error equalization using a priori information," *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 673–683, 2002.

[38] X. Yin, J. A. N. Goudriaan, E. A. Lantinga, J. A. N. Vos, and H. J. Spiertz, "A flexible sigmoid function of determinate growth," *Annals of Botany*, vol. 91, no. 3, pp. 361–371, 2003.

[39] T. L. Fine, *Feedforward Neural Network Methodology*, Springer, Berlin, Germany, 1999.

[40] F. Evennou and F. Marx, "Advanced integration of WiFi and inertial navigation systems for indoor mobile positioning," *Eurasip Journal on Applied Signal Processing*, vol. 2006, Article ID 86706, 2006.

[41] T.-R. Hsiang, Y. Fu, C.-W. Chen, and S.-L. Chung, "A MapReduce-based indoor visual localization system using affine invariant features," *Computers & Electrical Engineering*, vol. 39, no. 7, pp. 2369–2378, 2013.

[42] C. Yan, Y. Zhang, J. Xu et al., "Efficient parallel framework for HEVC motion estimation on many-core processor," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 12, pp. 2077–2089, 2014.

[43] C. Yan, Y. Zhang, J. Xu et al., "A highly parallel framework for HEVC coding unit partitioning tree decision on many-core processors," *IEEE Signal Processing Letters*, vol. 21, no. 5, pp. 573–576, 2014.

[44] C. Yan, Y. Zhang, F. Dai, X. Wang, L. Li, and Q. Dai, "Parallel deblocking filter for HEVC on many-core processor," *Electronics Letters*, vol. 50, no. 5, pp. 367–368, 2014.

[45] Y. Zhang, C. Yan, F. Dai, and Y. Ma, "Efficient parallel framework for H.264/AVC deblocking filter on many-core platform," *IEEE Transactions on Multimedia*, vol. 14, no. 3, pp. 510–524, 2012.

[46] C. Yan, Y. Zhang, F. Dai, and L. Li, "Highly parallel framework for HEVC motion estimation on many-core platform," in *Proceedings of Data Compression Conference (DCC '13)*, pp. 63–72, March 2013.