International Journal of Smart Home
Vol.10, No.10 (2016), pp.193-202
http://dx.doi.org/10.14257/ijsh.2016.10.10.18

Introducing Weighted Fingerprint Indoor Positioning

Jaegeol Yim[1]

Dept. of Computer Engineering, Dongguk University at Gyeongju, Gyeongbuk, 38066, KOREA
yim@dongguk.ac.kr

Abstract

The deployment of a fingerprint positioning method consists of an offline phase and a real-time phase. During the offline phase, a fingerprint database is built. This process is tedious and time consuming. Even so, the majority of WiFi-based indoor positioning methods implemented in practical indoor location–based service (ILBS) systems are the fingerprint method, because this method is accurate enough, whereas other WiFi-based indoor positioning methods are too inaccurate. During the real-time phase, the fingerprint method obtains a test fingerprint (a set of WiFi signals collected at that moment). Then, for each fingerprint, f_i, in the fingerprint database, it compares f_i with the test fingerprint in order to find the most similar f_i. This paper introduces a novel method of comparing two fingerprints in order to improve the accuracy of the fingerprint method. This method assigns weights to received signal strength indications (RSSIs) based on the variance of the RSSIs.

Keywords: *Indoor Location–Based Service, Indoor Positioning Method, WiFi, Fingerprint Positioning Method, Received Signal Strength Indication*

1. Introduction

When we consider only outdoor services, the location-based service industry seems to be in the ripening stage. However, the indoor location–based service (ILBS) industry seems to still be in its growing stage. As huge buildings and enormous underground shopping centers are built day by day, the demand for indoor location–based services will grow steadily. As a matter of fact, many indoor location–based service prototype systems have been introduced in recent research papers.

One of the essential techniques in developing ILBS systems is the indoor positioning technique. Among the many indoor positioning techniques, WiFi-based positioning techniques are the most attractive because WiFi is available in all huge buildings and shopping centers. Most of the WiFi-based indoor positioning methods belong to either the signal propagation–model group or the fingerprint group.

Even though the implementation procedure of a signal propagation–model indoor positioning method is simpler than that of the fingerprint method, the majority of WiFi-based indoor positioning methods implemented in practical ILBS systems use the fingerprint method because it is accurate enough, whereas the signal propagation model is too inaccurate. Making use of the variance in received signal strength indication (RSSI), this paper proposes a novel method to improve the accuracy of the fingerprint method.

[1] Corresponding Author

ISSN: 1975-4094 IJSH

2. Related Research

As man-made constructions get bigger and bigger, demand for ILBSs is increasing. Examples of ILBSs include museum tour guides and boarding reminders for air passengers who are far from their gate, to name only two [1].

Yang *et al*. [2] presented an iBeacon-based hospital guide system. The architecture of the system is three-layered: the perception layer, the network layer, and the application layer, as shown in Figure 1. A beacon is attached to every point of interest, including rooms, offices, and amenities in the hospital. Mobile devices like smartphones, personal digital assistants (PDAs), and tablets read beacon signals via Bluetooth. Mobile devices are connected to the server through WiFi.

The application layer provides a message push service and a navigation service to users. An example message is: "You are in the neural department of internal medicine. At present, three patients are waiting for service." The navigation service shows the shortest path from the user's current position to the destination, (the neural department, for example) on a floor map.

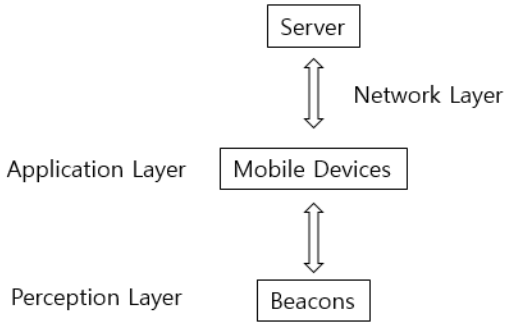

Figure 1. The Architecture of the iBeacon-based Hospital Guide System [2]

If the mobile terminal receives signals from at least three different access points (APs) of the wireless local area network (WLAN), then triangulation can be applied to estimate the position of the mobile terminal after converting the RSSIs into distances. Mahamud and Chowdhury [3] suggested that the RSSI from a cellular network (CN) should be used when the mobile terminal reads only two AP signals. The proposed algorithm is depicted in Figure 2. The algorithm finds the coordinates of the APs and the CN stations from a database.

Gu *et al*. [4] described the principle of LANDMARC, which is a popular indoor positioning system. LANDMARC places radio-frequency identification (RFID) tags in the area of the application (for example, exhibition rooms if the application is a museum guide), as shown in Figure 3. These tags are called reference tags. RFID readers are also installed in the application area. A moving object has an RFID tag, called a tracking tag, attached to it. The LANDMARC system collects RFID tag signals via the readers. Let the number of readers and reference tags be n (n=4 in Figure 3) and m (m=20 in Figure 3), respectively. For each reference tag, we may have n RSSIs read by the n readers. (If a reader is located too far from this reference RFID, then that reader cannot read the RSSI of the reference tag. We may assume the RSSI to be –MAXINT in this case.) Let $R_i = (R_{i1}, R_{i2}, \ldots R_{in})$ represent an n vector consisting of the RSSIs of the i-th RFID read by the n readers. The signal of the tracking tag is also read by the readers. Let $T = (T_1, T_2, \ldots T_n)$ be the RSSIs of the tracking tag read by the readers.

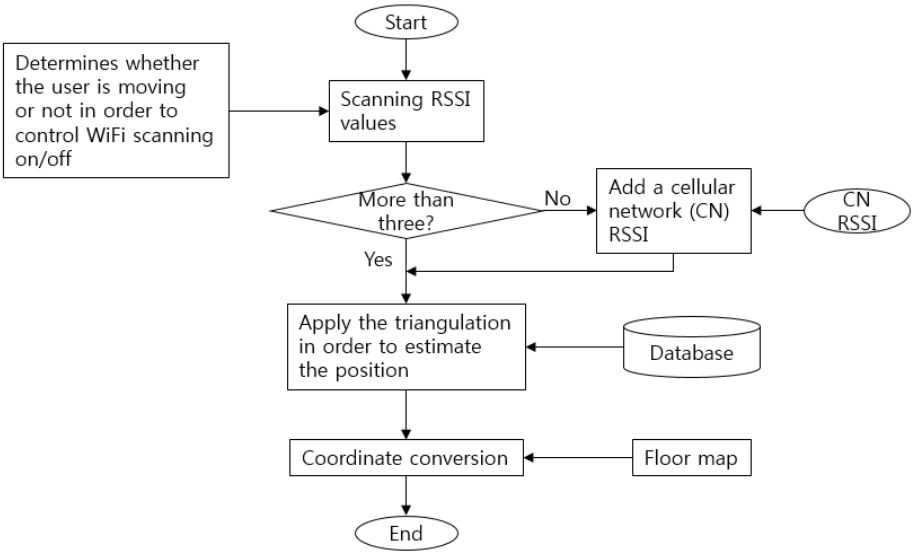

Figure 2. A Flowchart of the Indoor Positioning Algorithm Proposed by Mahamud and Chowdhury [3]

Figure 3. Configuration of a LANDMARC System [4]

Now, suppose that T is very close to R_i. What can we say about the current location of the moving object? We can conclude that the current location of the moving object is very close to the location of the i-th reference RFID in this case. The LANDMARC system selects the k-nearest reference RFIDs and lets the average of the coordinates of these k RFIDs be the current location of the moving object. When the average is calculated, the similarity between T and R_i is considered. For example, the inverse of the Euclidean distance between them can be used as the weight of the coordinates of the i-th reference RFID.

Given the k nearest neighbors (reference RFIDs), we can find the most centered one. Assuming this centered one is the tracking tag, we apply the positioning

procedure. Then the difference between the resulting coordinates and the real coordinates of the centered RFID can be considered the error of the positioning procedure. Making use of this error, Gu *et al*. [4] proposed a new algorithm, called an error self-correction algorithm.

Given the coordinates of the start point, we can calculate the coordinates of the destination point if we know the direction and the distance from the start point to the destination point. Yun *et al*. [5] attached an acceleration sensor and a geomagnetic sensor to a laptop. With a sequence of the acceleration sensor values, they counted the number of steps a pedestrian took from the start point to the destination point and converted the number of steps into the distance. With a geomagnetic sensor, they found out the direction from the start position to the destination. They performed experiments to determine a pedestrian's positions while walking around carrying a laptop equipped with these two sensors. They claimed that the average error of their positioning system is about 1.61 meters.

The positioning method used by Kim and Yim [1] is similar to the positioning method introduced by Yun *et al*. [5]. However, Yun *et al*. implemented their algorithm on a laptop, whereas Kim and Yim implemented the algorithm on a smartphone. Laptops have neither a built-in accelerometer nor a built-in geomagnetic sensor. On the other hand, smartphones have many built-in sensors, including an accelerometer and a compass. Therefore, the application introduced by Kim and Yim [1] can circulate easily.

The fingerprint positioning method based on RSSI is used as a common method in indoor positioning systems; it stores the previous signal pattern (a fingerprint) of each reference point (RP) in a database to compare that pattern with new, real-time signals (test fingerprints).

Ha *et al*. [6] found that the RSSI value obtained from an access point (AP) that is connected to a mobile terminal is significantly different from the RSSI value obtained from the same AP when it is disconnected from a mobile terminal. Based on this discovery, they claimed that using two suites of fingerprints will improve the accuracy of the fingerprint indoor positioning method. Then, they proposed the procedure for determining position depicted in Figure 4. It collects test fingerprints two times: one in the disconnected state and the other when connected.

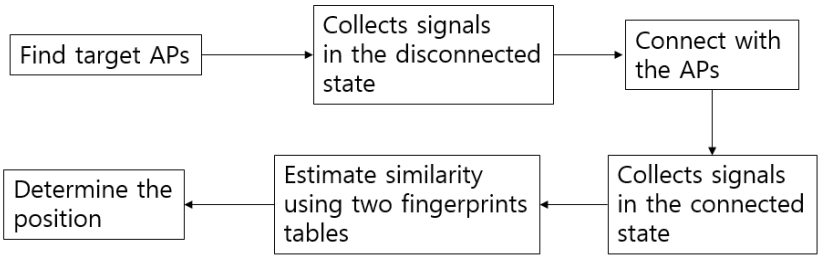

Figure 4. The Procedure for Determining Position Introduced by Ha *et al*. [6]

In the WiFi-based fingerprint indoor positioning process, we have to estimate the similarity between two fingerprints: a fingerprint from the fingerprint table and the test fingerprint. Traditionally, the Euclidean and Manhattan distances have been widely used in the measurement of signal similarity. The Euclidean distance from fingerprint $S_i = (S_{i1}, S_{i2}, \ldots S_{im})$ from the fingerprint table to test fingerprint $X = (x_1, x_2, \ldots x_m)$ is defined by:

$$Ed_i = \sqrt{(S_{i1} - x_1)^2 + (S_{i2} - x_2)^2 + \ldots + (S_{im} - x_m)^2}$$

The Manhattan distance from $S_i = (S_{i1}, S_{i2}, \ldots S_{im})$ to test $X = (x_1, x_2, \ldots x_m)$ is defined by:

$$Md_i = |S_{i1} - x_1| + |S_{i2} - x_2| + \ldots + |S_{im} - x_m|$$

After closely analyzing the RSSI pattern, So *et al*. [7] proposed the following expression to measure the distance from $S_i = (S_{i1}, S_{i2}, \ldots S_{im})$ to $X = (x_1, x_2, \ldots x_m)$:

$$d_i = \sqrt{C_{i1}^2 + C_{i2}^2 + \ldots + C_{im}^2}$$

$C_{ij} = S_{ij} - x_j \quad if |S_{ij} - x_j| \le D_{th}$, where D_{th} is a certain threshold

$C_{ij} = D_{th} \quad if |S_{ij} - x_j| > D_{th}$

RSSI-based indoor positioning methods can be categorized into two groups: signal propagation and fingerprinting. A signal propagation model shows the relationship between the RSSI and distance. With the distances from fixed nodes to a mobile terminal, we can estimate the location of the mobile terminal [8]. However, RSSI is not reliable due to multi-paths of the signal source and interference among signals [6]. Consequently, inaccuracy is the most significant shortcoming of signal propagation model–based approaches. Conversely, the fingerprinting methods are more accurate. However, this group requires a huge database of fingerprints. A fingerprint is a set of RSSIs collected at a certain location called a reference point. Collecting fingerprints is a tedious and time consuming job.

WLAN-based signal propagation–model indoor positioning is easy to implement. However, its performance is poor because of signal fluctuation. Li *et al*. [8] proposed a hybrid positioning approach. This method estimates a mobile terminal's position with the signal propagation model when the strength of the signal is greater than a certain threshold. When the strength of the signal is not greater than this threshold, the proposed system uses the fingerprinting method.

3. The Proposed Method

In the fingerprint database, we store the standard deviation (SD) in addition to the average of the RSSIs, as shown in Table 1. For each reference point, at least 30 fingerprints should be stored initially. This system allows users to voluntarily add more fingerprints to the database so it gets bigger and bigger as time goes by. We expect the accuracy of the system to improve as the database gets bigger.

Table 1. The Structure of the Fingerprint Database

	AP1	AP2	…	APn
RP1	-73	-85	…	-61
…	…	…	…	…
RP1	-78	-79	…	-68
Averages for RP1	-76.9	-83.2	…	-63.9
SDs for RP1	3.2	3.7	…	3.9
RP2	-82	-72	…	-91
…	…	…	…	…
RPm	-64	-88	…	-77

Table 2. The Structure of the Lookup Table

	AP1	AP2	AP3	AP4	…	APn
Averages for RP1	-76.9	-83.2	-38	-95.3	…	-63.9
SDs for RP1	3.2	3.7	1.0	10.5	…	3.9
Averages for RP2	-47	-89	-52	-79	…	-92
SDs for RP2	1.3	5.2	1.5	3.4	…	9.5
Averages for RP3	-44	-57	-91	-55	…	-54
SDs for RP3	0.9	1.7	9.0	1.6	…	1.6
Averages for RP4	-39	-81	-77	-42	…	-40
SDs for RP4	0.8	4.9	3.3	0.9	…	0.8
…	…	…	…		…	…
Averages for RPm	-95	-82	-56	-73	…	-97
SDs for RPm	10.1	3.3	1.6	3.1	…	14.2

This paper proposes the algorithm in Figure 5. Given a test fingerprint, $X = (x_1, x_2, \ldots x_m)$, obtained by a mobile terminal at the moment, this algorithm estimates the current position of the mobile terminal. This algorithm needs a lookup table consisting of m (the number of reference points) rows of averages and SDs in the database, as shown in Table 2. RPstructure is a record consisting of a point and a dissimilarity. A point is a pair of real numbers representing the X and Y coordinates of a reference point. RPstructure.dissimilarity is a real number representing the dissimilarity between RPstructure.point and the test fingerprint, X. The k-nearest reference points will be saved in the kRPs array of RPstructure. The largest dissimilarity in the kRPs is kept in the variable largest. The functions dissimilarity(), insert(), and average() are defined below.

```
CurrentPosition (LookupTable, vector X) {
    1. Define RPstructure consisting of a point (X, Y coordinates) and a real number
       (dissimilarity)
    2. RPstructure kRPs[k];  // k nearest RPs will be collected here
    3. Initialize kRPs.dissimilarity to MAXINT;  largest := MAXINT;
    4. (for each RP in LookupTable) {
    5.       thisDissimilarity = dissimilarity (X, RP);  // dissimilarity () is defined in Figure 6
    6.       insert(RP, thisDissimilarity);  // if thisDissimilarity is smaller than largest, then insert
       RP     into kRPs[]
    7. }
    8. Return average(kCPs); // average() is defined below
}
```

Figure 5. The Algorithm to Determine the Current Position of a Moving Object

Similar to the Manhattan distance function, our dissimilarity function calculates the difference between X[i] and RP[i]. However, there are two unique points in this algorithm. The first is that this algorithm assigns a weight, $(RP[i].SD)^{-t}$, where t is an arbitrary number such as 0.5, 2, 3,…, to the difference. X[i] or RP[i].average could be empty if the i-th access point is located too far from the mobile terminal or the reference point. If either X[i] or RP[i].average is empty, while the other is greater than a certain threshold, then X and RP cannot be similar. The second unique point is that this algorithm returns MAXINT in this case.


```
dissimilarity(X, RP) {
    1. thisDissimilarity := 0;
    2. for(i:=0; i<X.length; i++) {
    3.     if ((X[i] is empty and RP[i].average>th1) or (X[i]>th1 and RP[i].average is empty)
    4.         then return MAXINT;
    5.     thisDissimilarity += | X[i] − RP[i].average | / (RP[i].SD)^t
    6. }
    7. return thisDissimilarity;
}
```

Figure 6. The Algorithm to Calculate the Dissimilarity between Two RSSI Vectors

The parameters passed to the insert() function are a reference point and the dissimilarity between this reference point and the test fingerprint. If the dissimilarity is less than largest, then the insert() function inserts the reference point into the kRPs[] array and updates largest, as shown in Figure 7.

```
insert(RP, thisDissimilarity) {
    1. if (thisDissimilarity < largest) {
    2.   for (i:=0; i<k; i++) {
    3.     if(kRPs[i].dissimilarity == largest) {
    4.       kRPs[i].point := RP;
    5.       kRPs[i].dissimilarity := thisDissimilarity ;
    6.       break;
    7.     }
    8.   }
    9. } // End of insertion
       // update largest
   10. largest := kRPs[0].dissimilarity;
   11. for(i:=1; i<k; i++) {
   12.     if(kRPs[i].dissimilarity > largest) largest := kRPs[i].dissimilarity;
   13. }
}
```

Figure 7. Our Insert () Function

Our average() function returns the average of the k reference points in kRPs[]. This function uses the inverse of the dissimilarity for the weight, as shown in Figure 8. In the figure, the addition of two points, p1 and p2, returns (p1.X+p2.X, p1.Y+p2.Y).

```
average() {
    1. denominator, thisAverage := 0;
    2. (for i:=0; i<k; i++) {denominator += kRPs[i].dissimilarity^-u; }
    3. (for i:=0; i<k; i++) {thisAverage += kRPs[i].point * kRPs[i].dissimilarity^-u / denominator;
       }
    4. return thisAverage;
}
```

Figure 8. The Average() Function returns the Weighted Average of the Reference Points

Numerical Example 1: Suppose we have the LookupTable shown in Table 2, where n and m are all 5. Suppose further that we have X=(-60, -85, -44, -90, -87) and t is 2. Then, dissimilarity(X, RP1) is:

$$dissimilarity(X, RP1) = \frac{|-60 - -76.9|}{3.2^2} + \frac{|-85 - -83.2|}{3.7^2} + \frac{|-44 - -38|}{1^2} + \frac{|-90 - -95.3|}{10.5^2} + \frac{|-87 - -63.9|}{3.9^2}$$

= 9.348

Manhattan distance (X, RP1) = |-60 - -76.9| + |-85- -83.2| + |-44- -38| + |-90 - -95.3| + |-87- -63.9| = 53.1

Dissimilarity(X, RP2) = 12.4
Manhattan distance(X, RP2) = 41

Considering the Manhattan distance, we can conclude that RP2 is closer to X than RP1. However, the dissimilarity shows that RP1 is closer to X than RP2. Since the SD of AP3's RSSIs received at reference point 1 is very small (1.0), the difference between X[3] and RP1[3] dominates the dissimilarity.

If we set u=0.5, then we obtain

Dissimilarity(X, RP1) = 29.7

Dissimilarity(X, RP2) = 27.27

This result shows that RP2 is closer to X than RP1. From this example, we can conclude that we have to find the optimal t through experiment.

Numerical Example 2: Let X=(-50, -85, -50, -80, …), th1=-82, and let the LookupTable be Table 3. Note that X[5] is empty, and RP1[3] of LookupTable is also empty. The dissimilarity(X, RP1) is MAXINT, because X[3], which is -80, is greater than th1, and RP1[3] is empty.

Notice that dissimilarity(X, RP2) will put the designated small number, such as -500, in X[5] because X[5] is empty, while RP2[5] is not greater than th1. Notice also that RP2[5].average is very small, whereas RP2[5].SD is very big. We have these numbers because we replaced empty elements with -500s.

$$dissimilarity(X, RP2) = \frac{|-50 - -47|}{1.3^2} + \frac{|-85 - -89|}{5.2^2} + \frac{|-50 - -52|}{1.5^2} + \frac{|-80 - -79|}{3.4^2} + \frac{|-500 - -241|}{195^2}$$

Table 3. The Structure of Lookup Table

	AP1	AP2	AP3	AP4	…	AP5
Averages for RP1	-76.9	-83.2		-95.3	…	-63.9
SDs for RP1	3.2	3.7		10.5	…	3.9
Averages for RP2	-47	-89	-52	-79	…	-241
SDs for RP2	1.3	5.2	1.5	3.4	…	195
Averages for RP2	-44	-57	-91	-55	…	-54
…	…	…	…		…	…
Averages for RPm	-95	-82	-56	-73	…	-97
SDs for RPm	10.1	3.3	1.6	3.1	…	14.2

Numerical Example 3: Let kRPs[] be [((300, 300), 30), ((400, 200), 20), ((500, 100), 20)], and let u be 2. The denominator and thisAverage will be:

$$denominator = \frac{1}{30^2} + \frac{1}{20^2} + \frac{1}{20^2} = 0.006111$$

$$AverageX = (300)*\left(\frac{30^{-2}}{deno\ minator}\right)+(400)*\left(\frac{20^{-2}}{deno\ minator}\right)$$
$$+(500)*\left(\frac{20^{-2}}{deno\ minator}\right)=422.7$$

$$AverageY = (300)*\left(\frac{30^{-2}}{deno\ minator}\right)+(200)*\left(\frac{20^{-2}}{deno\ minator}\right)$$
$$+(100)*\left(\frac{20^{-2}}{deno\ minator}\right)=177$$

$$thisAverage = (300,300)*\left(\frac{30^{-2}}{deno\ minator}\right)+(400,200)*\left(\frac{20^{-2}}{deno\ minator}\right)$$
$$+(500,100)*\left(\frac{20^{-2}}{deno\ minator}\right)=(422.7,177)$$

As the dissimilarity of reference point (500, 100) is less than that of reference point (300, 300), the average should be closer to (500,100).

4. Conclusions

After reviewing existing location-based service systems and indoor positioning techniques, this paper introduced a new algorithm to determine the current position of a moving object. The algorithm is a WiFi-based fingerprint indoor positioning algorithm that uses the standard deviation of RSSIs from an access point as the weight of the AP. The algorithm is a kind of k-nearest neighbor algorithm, because it selects the k-nearest reference points. This algorithm uses the dissimilarity between the fingerprint of the reference point and the test fingerprint as the weight when it calculates the average of the k-nearest reference points. Given a test fingerprint, X, and the fingerprint of a reference point, RP, this algorithm returns MAXINT if X[i] is greater than a certain threshold while RP[i] is empty. Making use of the proposed algorithm, we are developing a practical location-based mobile application.

Acknowledgments

This research was supported by the Dongguk University Research Fund of 2016, by the Ministry of Education (NRF-2011-0006942), and by the Ministry of Knowledge Economy (10037393).

References

[1] S. Kim and J. Yim, "A Survey of Fingerprint Indoor Positioning Techniques," Asia-pacific Proceedings of Applied Science and Engineering for Better Human Life, Vol. 9 **(2016)**, pp. 80-83.
[2] J. Yang, Z. Wang and X. Zhang, "An iBeacon-based Indoor Positioning Systems for Hospitals," IJSH Vol. 9, No.7, **(2015)**, pp.161-168.
[3] J. Yang, Z. Wang and X. Zhang, "An iBeacon-based Indoor Positioning Systems for Hospitals," IJSH Vol. 9, No.7, **(2015)**, pp.161-168.
[4] M. Mahamud and M. Chowdhury, "Indoor Location System with Wi-Fi and Alternative Cellular Network Signal," IJMUE Vol.10, No.3, **(2015)**, pp.59-70.
[5] Y. Gu, J. Zhang, J. Wang, B. Gao and J. Du, "RFID Indoor Localization Algorithm Based on Adaptive Self-correction," IJSH Vol. 8, No.6, **(2014),** pp.205-216.
[6] K. Yun, Y. Jo, N. Kim, U. Jo, S. Yang and Y. Kim, "A Practical Indoor Position Estimation by Using a Laptop Computer Equipped With Sensors," IJSH Vol. 7, No.4, **(2013)**, pp.27-36.

[7] I. Ha, Z. Zhang and C. Kim, " Improving Accuracy of FingerPrint DB with AP Connection States," IJSEIA Vol. 6, No.2, **(2012),** pp.143-148.
[8] J. So, J. Lee, C. Yoon and H. Park, "An Improved Location Estimation Method for Wifi Fingerprint-based Indoor Localization," IJSEIA Vol. 7, No.3, **(2013)**, pp.77-86.
[9] J. Li, B. Zhang, H. Liu, L. Yu and Z. Wang, "An Indoor Hybrid Localization Approach Based on Signal Propagation Model and Fingerprinting," IJSH Vol. 7, No.6, **(2013),** pp. 157-170.

Author

Jaegeol Yim, He received an MSc and PhD in Computer Science from the University of Illinois at Chicago, in 1987 and 1990, respectively. He is a Professor in the Department of Computer Engineering at Dongguk University at Gyeongju Korea. His current research interests include Petri net theory and its applications, location-based services, AI systems, and multimedia systems. He has published more than 50 journal papers, 100 conference papers (mostly written in the Korean Language), and several undergraduate textbooks.

