International Journal of Grid and Distributed Computing
Vol. 5, No. 1, March, 2012

A Review of the Techniques for Indoor Location based Service

Gyeyoung Lee and Jaegeol Yim

Dongguk University at Gyeongju, Korea
{lky, yim}@dongguk.ac.kr

Abstract

A location based service (LBS) provides useful information to the users based on the geographic location the user designates or the user is currently located. Examples of LBS include directory service, gateway service, location utility service, presentation service, route service, and so on. These services are very useful to the users, and LBS should be available indoor: subways, large shopping malls, department stores, factories, and so on. This manuscript summarizes the techniques needed in development of indoor LBS (ILBS) including indoor positioning, indoor moving objects database system, rendering drawings, and web service.

Keywords: *Indoor Location Based Service, Indoor Positioning, Fingerprints, K-NN, Web Services, Kalman filter*

1. Introduction

A location based service (LBS) is to provide useful information to the users based on the geographic location the user designates or the user is currently located. Examples of LBS include directory service, gateway service, location utility service, presentation service, route service, and so on [1]. These services are very useful to the users and should be provided not only outdoor but also indoor. Those LBSs provided indoor is called ILBS (Indoor Location Based Service). However, most of the existing LBSs are outdoor ones and they use GPS to determine user's current location. In the case of indoor, GPS signal cannot be utilized and there is no economic way of determining user's location even though there are many research results about indoor positioning have been published.

This manuscript surveys the fundamental techniques for ILBS development including positioning, database management, rendering drawings, and web services. LBS cannot be realized without the information of user's location positioning, determining user's location, is the most important and essential ingredient of LBS. Database management is another important ingredient because there are many moving, pedestrians for example, and stationary, stores for example, objects involved in an ILBS system. Rendering drawings is another important research subject in the field of ILBS because a drawing is the most important component of the user interface of an ILBS system. For LBS, the map is used as the main component of the user interface, but in the case of ILBS, a drawing is an unpublished private property. The solutions for those problems of positioning, database management and rendering drawing should be published as web services so that other application developers can utilize them in their development.

2. Related Works

There are many positioning systems including GPS [2], wide-area cellular-based systems [3], infrared-based system [4], radio frequency (RF) + ultrasonic-based systems [5, 6], physical contact systems [7], various computer vision systems [8], and RF based systems [9, 10]. Among them RF-based wireless LAN positioning techniques are most interesting in the field of indoor positioning because of the following reasons. GPS signal is not available inside of a building so GPS system cannot be an indoor positioning system and the others require special equipments dedicated for positioning. Whereas RF-based wireless LAN positioning systems do not require additional hardware dedicated for positioning and wireless network is being serviced everywhere including college campuses, airports, hotels and homes [11].

A moving object database (MODB) is a database representing information on moving objects, particularly their locations. The purpose of the MODB is to provide answers to various spatiotemporal information queries such as the following: “Where am I?” “Who is the closest friend to me?” “How far is he?” “When and where did I meet Kim?” A location-based service (LBS), such as the fleet management system for a taxi company or a logistics company, cannot be realized without the availability of spatiotemporal information. Therefore, MODB techniques have been developed in the LBS field.

For most MODB applications, the moving object is equipped with a GPS (Global Positioning System) unit that sends information on the object’s location, time, and velocity, among others, to the MODB. However, GPS signals are usually unavailable inside enclosed structures; thus, MODB techniques used for outdoor LBSs cannot be directly applied to indoor LBS systems. An indoor moving object can be a mini-notebook computer, a PDA, or a smart phone carried by a pedestrian. The physical care covered by indoor moving objects is much smaller than that by outdoor ones, and the speed of indoor moving objects is much slower than that of outdoor ones.

Indoor LBSs are much more useful than outdoor LBSs. Indoor LBSs may include an electronic museum guide, a shopping concierge at a department store, asset tracking, VoIP-911 caller location identification, and so forth. An electronic museum guide pushes the digital content related to the exhibit that the user is viewing, suggests points of interest, and provides the information the user requests [12]. The following summary is also from [12].

In the field of MODB, models of location information, techniques of uncertainty management, query languages accessing spatiotemporal data, indexing techniques, data mining, and security have to be studied. Wolfson has reviewed these topics in [13]. The location information in a MODB is naturally uncertain. Olston introduced a method of handling the uncertainty [14] and Tao et al. introduced “probably restricted rectangle” and the information retrieval method using the rectangle [15].

The positions of the spacecrafts are important information in national defense system. [16] introduces their design of moving object database system for military command. The main elements of the system include a traditional relational DBMS, spatial wrapper of the DBMS, API component between GUI and the DBMS. [17] is another MODB prototype implemented on top of a traditional relational DBMS. [18] proposes a CASE tool supporting spatiotemporal conceptual modeling for moving object database applications. It has the following two advantages: (1) It supports both spatiotemporal and non-spatiotemporal characteristics and compatible with the ER model. (2) It supports systematic spatiotemporal characteristics and can represent different spatiotemporal changes for diverse moving objects applications. [19] explains how to use SECONDO in building a moving objects database system.

The query based indexing techniques are about indexing queries of the moving objects and not about indexing the location of these objects. The location based indexing techniques can be divided into two categories, indexing the current and future positions and indexing the past positions. These indexing methods are all variants of R-Tree. [20] introduces Delineated R-Tree (DR-Tree) indexing structure which is highly balanced and has performance advantages over other R-tree based indexing methods. [21] proposes another new indexing method, S-TB Tree. [22] proposes the indexing method of efficiently processing historical spatiotemporal pattern queries,

There is no commercial moving object database in the markets yet. Applications of MODB inevitably provide a map on their user interface. [23] introduces a digital map to indicate locations of interested static or moving objects. They use Oracle Spatial SDOAPI Java Class Library to handle spatial data. [24] presents a formal model where the geometric components of the thematic layers in a GIS are represented as an OLAP(On Line Analytical Processing) dimension hierarchy and introduces the notion of spatial aggregation. Then, they also address moving object aggregation over a GIS.

The advances in Global Positioning Systems, wireless communication systems and miniaturization of computing devices have brought an emergence of various applications in Location-Based Services (LBS). As a result, there is an increasing need for efficient management of vast amounts of location-in-time information for moving objects. Therefore, one of the most important issues a MODB must address is spatio-temporal languages. The queries MODB should be able to handle include finding the similar trajectories of a moving object. With this kind of query, we can determine migration patterns of animals and recommend a route to a driver. Nearest Neighbor (NN) query is defined as follows: Given a set of trajectories T and a trajectory Q, find the trajectory in T which has the smallest distance to Q. If we find 2 closest trajectories instead of just 1 then the query is called 2NN. [25] addresses Set Nearest Neighbor query. An example KNN(K-nearest neighbor) query is “What will be the query’s 3NN 2 minutes from now?” [26] introduces a new multi-threaded and cache-conscious algorithm to efficiently process massive continuous KNN queries. [27] proposes techniques including pruning candidate objects for processing a nearest neighbor query when the location of the query object is specified by an imprecise Gaussian distribution. [28] proposes a Frechet distance based approach to similarity join for large sets of moving object trajectories and shows that the proposed method is 50% faster than the traditional methods. There are two kinds of moving objects: moving points like vehicles and moving regions like hurricanes and oil spills. Spatial objects evolve as time flows and their topological relationships develop over time. Spatio-temporal predicates have been proposed to ask for these time-varying relationships. [29] proposes a generic algorithmic scheme that can evaluate spatio-temporal predicates. The following is an example continuous spatio-temporal query: “Continuously, inform commander C with all friendly units that are within ten miles from soldier S.” To minimize the execution cost of this kind of queries, [30] proposes Query-Track-Participate(QTP) query processing model where a query is continuously answered by a querying server, a tracking server, and a set of participating servers. [31] introduces an algebraic Spatio-Temporal Trajectory data type (STT) for the representation of object trajectory. The STT is an abstract data type endowed with a set of operations designed as a way to cover the syntax and semantics of a given trajectory.

Updating policies for MODB are thoroughly discussed in [32]. They introduced three moving object database updating policies, Speed Dead-Reckoning (sdr), adaptive dead reckoning (adr), and disconnection detecting dead-reckoning (dtdr). A dead-reckoning update policy updates the moving object's database location only when the difference between its actual location and its database location is greater than the threshold th. The threshold th is a constant during the trip in the case of sdr, whereas it is newly computed whenever the database location is updated in the case of adr. In the case of dtdr, the threshold th is not only newly computed whenever the database location is updated but it also decreases as the time interval since the last update increases.

Typical applications of MODB include intelligent transport systems, digital battlefield, location e-commerce, and so forth. A query that must be evaluated for a time interval, "List the nearest gas stations to my vehicle in the next 15 minutes" for example, is called a continuous query. Existing MODBs work well for instantaneous queries but not for continuous ones. [33] proposes a new updating strategy which improves the quality of continuous query results. Then, it provides their simulation results to show the effectiveness of their updating strategy. In their simulation, they use a standardized random number generator to assign velocity to the moving object. This implies that they assume that a moving object is equipped with a device like GPS which provides the velocity of the moving object. In the case of indoor positioning there is no device which provides the velocity of the moving object.

The estimated values are used to predict the moving object's future position. For the future prediction, most techniques use some mathematical formulas of motion derived from its recent movements. However, an object's movements are not that simple. Therefore, an object's trajectory patterns are used for prediction recently. [34] introduces a hybrid prediction model where both mathematical formulas and trajectory patterns are considered. [35] proposes an asynchronous position updating algorithm for continuous queries of moving objects which improves accuracy, reduces the communication cost, and balances server load more evenly. An example of continuous query is "All vehicles which will appear in the region R in the next 10 minutes."

Processing the location stream which is faster than the database can handle is one of the problems we have to solve in order to build a high performance MODB. [36] proposes a location filter in order to address the problem of processing the location stream. The location filter consists of the on-line location filter and the temporal location manager. The on-line location filter executes the filtering algorithm plug-ins and filters the location stream on-line while the temporal location manager insert location data into the database at an acceptable insertion rate.

Techniques of rendering and manipulating a map have been developed in the field of GIS. Nowadays, many organizations provide open APIs for map manipulation. However, these are all for outdoor LBS development [37, 38]. This manuscript introduces the module of rendering drawings we have implemented.

Web services are typically application programming interfaces or Web APIs that are accessed via Hypertext Transfer Protocol and executed on a remote system hosting the requested services [39]. This manuscript introduces some web services providing essential functions for ILBS development.

3. Our Research Results

In this section, the techniques we have developed will be introduced. The materials in this section are from the papers we have published.

3.1. Indoor Positioning

3.1.1. K-Nearest Neighbor Method: In K-NN, we build a look-up table in the first phase, or off-line phase. The entire area is covered by a rectangular grid of points called *candidate points*. At each of candidate points we measure the RSSs many times. Let RSS_{ij} denote the j-th received signal strength of the signal sent by AP_i. A row of the look-up table is an ordered pair of (coordinate, a list of RSSs). A coordinate is an ordered pair of integers (x, y) representing the coordinates of a candidate point. A list of signal strengths consists of five integers, RSS_1, RSS_2 ..., where RSS_i is an average of signal strengths RSS_{ij} received at (x, y) and sent by AP_i. An example of look-up table is shown in Table 1.

In the second phase, or on-line phase, the positioning program gathers RSSs the user receives at the moment. If the positioning program is running on the user's handheld terminal, then the terminal itself will collect RSSs. Let X=(-40, -56, -54, -69, -66) be the vector of the collected RSSs. K-NN, then look up the look-up table and finds the closest candidate point, CP2 in the case of Table 1, and returns it as the user's current location. If K equals 2, then it will find two closest candidate points and return the average of them as the user's current location.

Table 1. An Example Look-up Table of K-NN (C.P stands for Candidate Points, CP_i is coordinates of i-th C.P, APi is the MAC address of i-th AP)

AP / C.P	AP1	AP2	AP3	AP4	AP5
CP1	-39	-55	-56	-70	-67
CP2	-40	-56	-55	-69	-66
CP3	-44	-42	-62	-45	-61
...	...	...	...	...	...

3.1.2. Decision Tree for Positioning: In the off-line phase of decision tree method, we build a decision tree with training data. An example training data set is shown in Table 2. Table 2 is similar to Table 1. Only differences are 1) RSSs are not averages, 2) RSSs are discretized. An example of discretizing policy can be I1 = {x| x>-30}, I2 = $\{x| -40 < x \le -30\}$, I3 = $\{x| -50 < x \le -40\}$, ...

Given a set of training data, we build a decision tree with the algorithm Construct_DT shown in Fig. 1. Step (4) of the algorithm computes *I* where *I* is the expected information needed to classify a given sample and is given by

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

where, m is the number of candidate points, S is the number of tuples in the training data set (rows of Table in the algorithm), s_i is the number of rows of S in class CP_i, $p_i = {s_i}/{S}$.

Step (5) of the algorithm computes the entropy, or expected information based on the partitioning into subsets by CP_i. Let CP_i have v distinct values, $\{a_1, a_2, ..., a_v\}$. CP_i can be used to partition S into v subsets, $\{S_1, S_2, ..., S_v\}$, where S_j contains those samples in

S that have value a_j of CP_i. Let S_{ij} be the number of samples of class CP_i in a subset S_j. The entropy E(CP_i) is given by

$$E(CP_i) = \sum_{j=1}^{v} \frac{s_{ij} + \ldots + s_{mj}}{s} I(s_{ij}, \ldots, s_{mj}).$$

At step (6) the algorithm computes information gain G(CP_i) by the following expression, $Gain(CP_i) = I(s_1, s_2, \ldots, s_m) - E(CP_i)$.

Table 2. An Example Training Data Tuples (C.P stands for Candidate Points, CP_i is coordinates of i-th C.P, APi is the MAC address of i-th AP, I stands for interval)

AP / C.P	AP1	AP2	AP3	AP4	AP5
CP1	I2	I1	I2	I5	I5
	I1	I1	I2	I1	I1
	...				
CP2	I3	I2	I3	I1	I2
	...				
...	...				

An example decision tree is shown in Figure 1. Depth of a decision tree is the number of APs, or N, and the number of branches of a node is the number of intervals, or I. A leaf of a decision tree is labeled with CP_i, or candidate point. Given a vector of measured RSSs, X=(RSS1, RSS2, ...), the on-line phase of decision tree method, when the decision tree built in the off-line phase looks like the one in Fig. 1, examines RSS2 because the root of the tree is labeled AP2. If RSS2 belongs to I2, then it will take the second branch and will examine RSS4 since the second child node of the root is labeled AP4. For each node, there are at most I branches and the depth of a decision tree is N. Therefore, the time complexity of the on-line phase of decision tree method is *O(I*N)*.

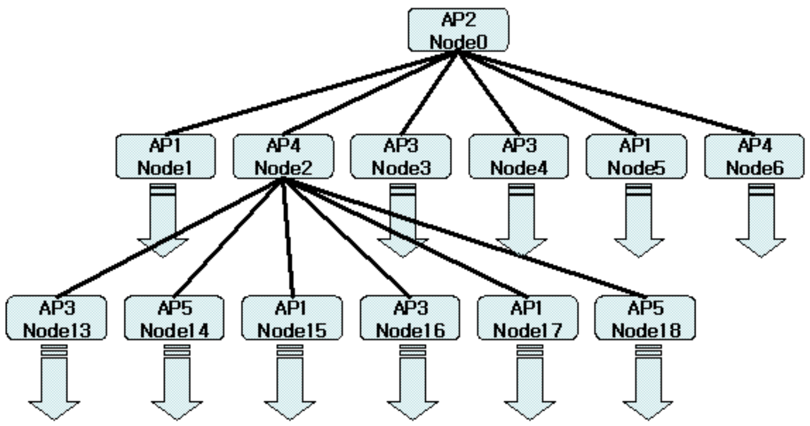


Figure 1. An Example Decision Tree where I=6, N=5, and M=96.

Experimental Evaluation

For experimental evaluation of accuracy of the methods, we have implemented K-NN, Bayesian and decision tree methods on a SAMSUNG SENS R50 laptop computer equipped with Intel(R) PRO/Wireless 2200BG Network Connection as its WLAN card using Microsoft Visual C# 2005 Express Edition Beta. It is known that neural network gets worse accuracy than K-NN and we did not implement neural network method. The experimental results are presented in this section.

Test Bed

The test bed for the experiments is the Micro LAB on the 4-th floor of Natural Science Building as shown in Figure 2.

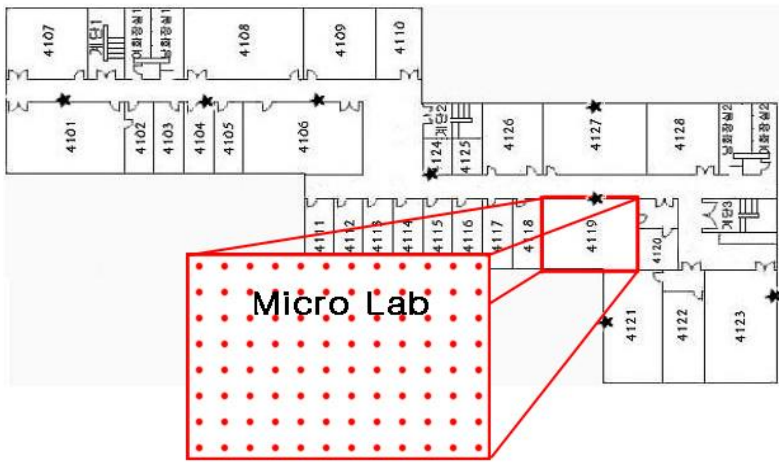


Figure 2. The Test Bed

Test Results

We first performed experiments to see if the number of samples for one entry (the average of the samples) of Lookup table is related to the accuracy of K-NN. With the results shown in Figure 3, we decided to use the average of 30 samples as an entry of Lookup table. We have used X to denote the vector of RSSs measured in the on-line phase. We obtained X with the average of 10 measures for this experiment.

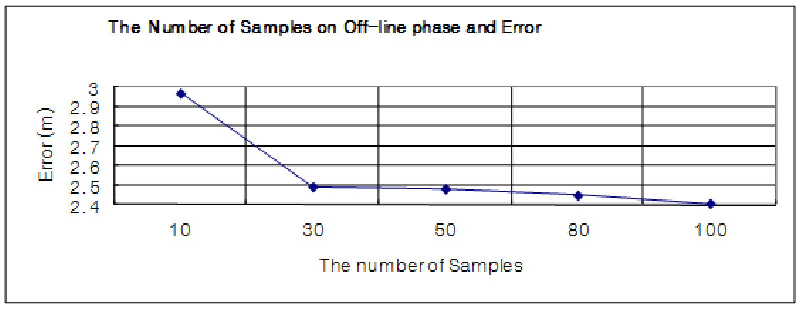


Figure 3. The Number of Samples for an Entry of Lookup Table and Accuracy

We have used X to denote the vector of RSSs measured in the on-line phase. We performed experiments to see how the number of samples for X is related to accuracy. Figure 4 shows the result of the experiments. An entry of the lookup table for this experiment was the average of 100 samples. With the result we can conclude that if we use the average of 10 samples as X then the accuracy will be much improved.

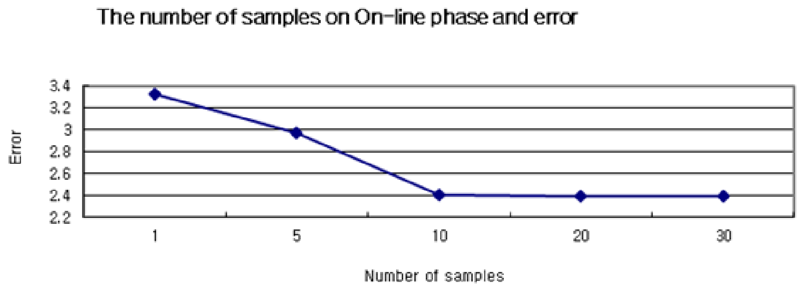


Figure 4. The Number of Samples for X and Accuracy

We performed experiments of running 1-NN, Bayesian and decision tree methods on training data of N=5, I=6, and M=96 in order to compare their accuracies. The test results are shown in Figure 5. In the figure, 'number of samples' is the same as the number of measurements we have performed for each entry of Lookup table. An entry of Lookup table is the average of the measurements. When the number of samples is 10, K-NN method is much better than others. However, the difference gets decreases as the number of samples increases and when the number of samples is 50 the accuracies of the three methods are almost same. For this experiment, X was the average of 10 measurements.

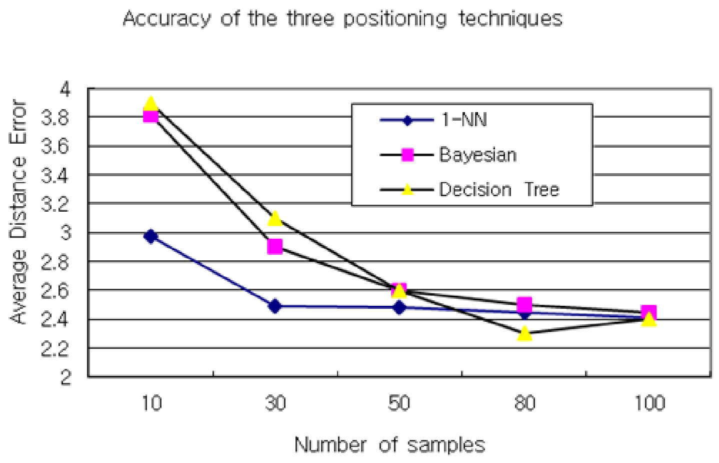


Figure 5. Accuracy of Three Methods

3.1.3. Trilateration: If we measure N ranges, $r_1, r_2, \ldots, r_N$ from N base stations, $n_1 = (X_1\ Y_1\ Z_1)^T,\ \ldots, n_N = (X_N\ Y_N\ Z_N)^T$ to a mobile terminal, $m = (x\ y\ z)^T$ as shown in Figure 6, then we can estimate the coordinates of m by using trilateration. By squaring, r_i^2 can be expressed as follows:

$$(x-X_i)^2+(y-Y_i)^2+(z-Z_i)^2=r_i^2,\ (\,for\ i=1,2,...,N)$$

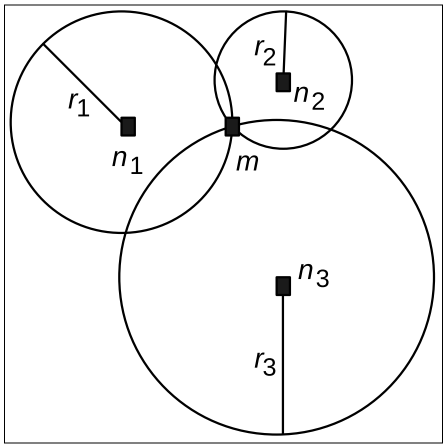


Figure 6. A Diagram to Illustrate Trilateration

By subtracting r_1^2 from $r_i^2(i=2,\ldots,N)$, we have $A\vec{x}=\vec{b}$, where

$$A=2\begin{bmatrix}(X_2-X_1) & (Y_2-Y_1) & (Z_2-Z_1)\\ \vdots & \vdots & \vdots\\ (X_N-X_1) & (Y_N-Y_1) & (Z_N-Z_1)\end{bmatrix},\ \vec{x}=\begin{bmatrix}x\\ y\\ z\end{bmatrix}$$

$$\vec{b}=\begin{bmatrix}(X_2^2-X_1^2)+(Y_2^2-Y_1^2)+(Z_2^2-Z_1^2)-(r_2^2-r_1^2)\\ \vdots\\ (X_N^2-X_1^2)+(Y_N^2-Y_1^2)+(Z_N^2-Z_1^2)-(r_N^2-r_1^2)\end{bmatrix}.$$

When the coordinates are 3 dimensional, we need to have at least 4 base stations. By applying the MMSE (Minimum Mean Square Error) method, we can obtain $\hat{\vec{x}}$ (approximation of $\vec{x}$) with the following position estimates:

$\hat{\vec{x}}=(A^TA)^{-1}A^T\vec{b}$ --------------- (Expression 1)

3.1.4. Extended Kalman Filter: The Kalman filter iteratively predicts the position of the mobile terminal and updates the prediction with new measurements to yield an estimate. In positioning, the measurement equation is represented as a nonlinear model, and linearization should be performed to derive a linear equation. The extended Kalman filter (EKF) considers the real-time linearization of the system function at the previous state estimate and that of the observation function at the corresponding predicted position. The measured distances, r_i, can be expressed as follows:

$$r_i=\sqrt{(X_i-x)^2+(Y_i-y)^2+(Z_i-z)^2}+v_i$$

where v_i represents the measurement noise and is assumed to be white Gaussian noise (AWGN) with a normal probability distribution of 0 mean and σ_i^2 variance. Using the Taylor approximation at a nominal point $\vec{x}_0 = (x_0 \ \ y_0 \ \ z_0)^T$, we have:

$$r_i - r_{i0} = \left.\frac{\partial r_i}{\partial \vec{x}}\right|_{\vec{x}=\vec{x}_0} \delta\vec{x} + v_i = h_x^i \delta x + h_y^i \delta y + h_z^i \delta z + v_i$$

where $r_{i0} = \sqrt{(X_i - x_0)^2 + (Y_i - y_0)^2 + (Z_i - z_0)^2}$ is the computed distance between the nominal point and the i-th base station, $(h_x^i = \frac{x_0 - X_i}{r_{i0}}, h_y^i = \frac{y_0 - Y_i}{r_{i0}}, h_z^i = \frac{z_0 - Z_i}{r_{i0}})$ is the LOS (Line Of Sight) vector from the base location to the i-th base station, and $\delta\vec{x} = (\delta x \ \ \delta y \ \ \delta z)$ is the position error vector to be determined. For N base stations, we have the following expression:

$$\begin{bmatrix} r_1 - r_{10} \\ \vdots \\ r_N - r_{N0} \end{bmatrix} = \begin{bmatrix} h_x^1 & h_y^1 & h_z^1 \\ \vdots & \vdots & \vdots \\ h_x^N & h_y^N & h_z^N \end{bmatrix} \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix} + \begin{bmatrix} v_1 \\ \vdots \\ v_N \end{bmatrix}$$

This equation can be rewritten in the following form, where subscript k is the index for the discrete time sequence:

$$\delta\vec{r}_k = \vec{r}_k - \vec{r}_0 = H_k \delta\vec{x}_k + \vec{v}_k \quad \text{----------- (Expression 2)}$$

The measurement noise is AWGN with $\vec{v}_k \sim N(0, R_k)$. Here, $R_k = diag(\sigma_i^2)$ where *diag* stands for diagonal matrix. If we have more than 3 measured distances, we can estimate the position of the mobile terminal by using the WLSE (Weighted Least Squares Estimate). However, by adding system models to the above measurement models and applying the Kalman filtering process, a more reliable position can be found. The P (Position), PV (Position Velocity) and PVA (Position Velocity Acceleration) models are generally used in navigation as a system model. For positioning, we can use the P model described by the following expression:

$$\vec{x}_{k+1} = \Phi_k \vec{x}_k + \vec{w}_k$$

where Φ_k is the state transition matrix and $\vec{w}_k \sim N(0, Q_k)$ the modeling error, or the system error. Table 3 summarizes the EKF processing. The estimate we can obtain with these expressions is the value of $\delta\hat{\vec{x}}_k$, and the final solution, i.e. the user's location, can be obtained with the following expression:

$$\hat{\vec{x}}_k = \vec{x}_0 + \delta\hat{\vec{x}}_k .$$

Table 3. A Summary of the EKF Processing

Linearized state model: $\vec{x}_{k+1} = \Phi_k \vec{x}_k + \vec{w}_k, \vec{w}_k \sim N(0, Q_k)$
Linearized measurement model: $\vec{r}_k = \vec{r}_0 + H_k \delta \vec{x}_k + \vec{v}_k, \vec{v}_k \sim N(0, R_k)$
1) Initial guess: $\vec{x}_0^- = E(\vec{x}_0)$ and $P_0^- = \mathrm{var}(\vec{x}_0)$
2) Linearizing:
$\vec{r}_k = \vec{r}_0 + H_k \delta \vec{x}_k + \vec{v}_k, \vec{r}_0 = \begin{bmatrix} \sqrt{(X_1 - x_k^-)^2 + (Y_1 - y_k^-)^2 + (Z_1 - z_k^-)^2} \\ \vdots \\ \sqrt{(X_N - x_k^-)^2 + (Y_N - y_k^-)^2 + (Z_N - z_k^-)^2} \end{bmatrix}$
3) Kalman Gain: $K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1}$
4) Measurement update: $\hat{\vec{x}}_k = \hat{\vec{x}}_k^- + K_k(\vec{r}_k - \vec{r}_0)$
5) Update error covariance: $P_k = (I - K_k H_k) P_k^-$
6) State propagation: $\hat{\vec{x}}_{k+1}^- = \Phi_k \hat{\vec{x}}_k$, $P_{k+1}^- = \Phi_k P_k \Phi_k^T + Q_k$
7) Go to step 2

Positioning is a special case of the EKF process where $\Phi_k = I$ and $Q_k = 0$. It replaces step 6 in Table 3 by the following simplified expression:

$$\hat{\vec{x}}_{k+1}^- = \hat{\vec{x}}_k, \ P_{k+1}^- = P_k$$

The implementation of a positioning system using trilateration or the extended Kalman filter requires the relationship between the distances and RSSIs. For this purpose, we read the RSSI every 1m from an AP 300 times and, based on this information, plotted the relationship between the distance and RSSI, as shown in Figure 7. Using this relationship, we can obtain the distance to the AP based on the RSSI from the AP.

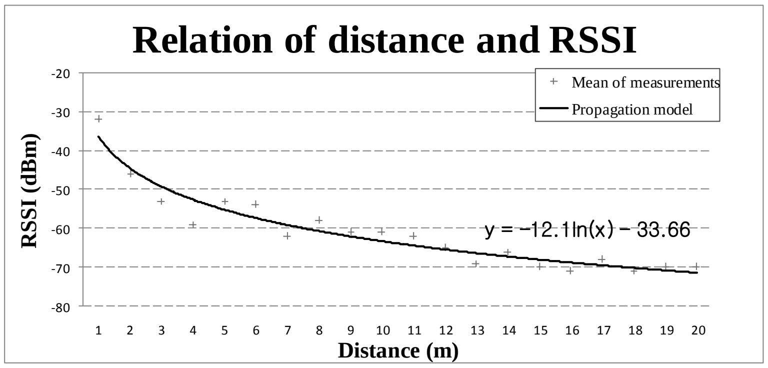
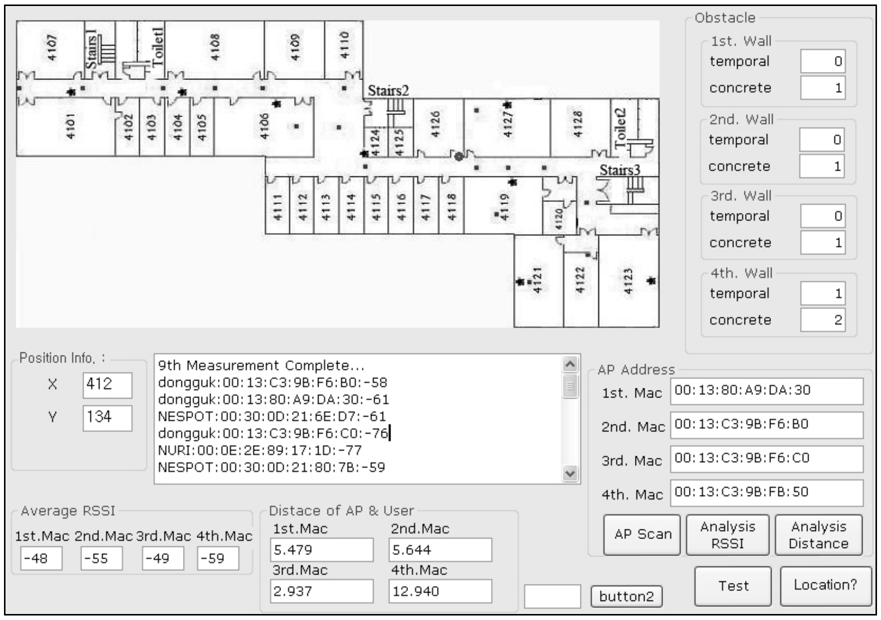


Figure 7. Relation between Distance and RSSI

Experimental Analysis

The graphical user interface (GUI) of the trilateration-based positioning system is shown in Figure 8. The floor plan shown in Figure 8 represents the 4th floor of the Natural Science Building, the test bed. When the button, “Location?”, is clicked, the system runs the trilateration algorithm and writes the X and Y coordinates returned from the algorithm in the box labeled “Position Info :” It also draws a big dot at the position (X, Y) on the floor plan. When the button, “Test”, is clicked, it provides dialog boxes in which the actual user’s location and the integer N can be input. It then runs the trilateration algorithm N times and records the results in the file named Result. We also implemented the EKF positioning method, and its GUI is similar to that of trilateration-based positioning system shown in Figure 8.



Obstacle

1st. Wall	
temporal	0
concrete	1

2nd. Wall	
temporal	0
concrete	1

3rd. Wall	
temporal	0
concrete	1

4th. Wall	
temporal	1
concrete	2

Position Info. :
X 412
Y 134

```
9th Measurement Complete...
dongguk:00:13:C3:9B:F6:B0:-58
dongguk:00:13:80:A9:DA:30:-61
NESPOT:00:30:0D:21:6E:D7:-61
dongguk:00:13:C3:9B:F6:C0:-76
NURI:00:0E:2E:89:17:1D:-77
NESPOT:00:30:0D:21:80:7B:-59
```

AP Address

1st. Mac	00:13:80:A9:DA:30
2nd. Mac	00:13:C3:9B:F6:B0
3rd. Mac	00:13:C3:9B:F6:C0
4th. Mac	00:13:C3:9B:FB:50

Average RSSI

1st.Mac	2nd.Mac	3rd.Mac	4th.Mac
-48	-55	-49	-59

Distace of AP & User

1st.Mac	2nd.Mac	3rd.Mac	4th.Mac
5.479	5.644	2.937	12.940

Figure 8. A Typical GUI of the Positioning Systems

There are many parameters affecting the efficiency of a positioning system and the number of APs with significant signal strengths is one of them. If the RSSI from an AP is less than -70 then the distance to the AP cannot reliably be determined and we consider it to be insignificant. In the following experiments, we had 4 APs with significant signal strengths.

They also performed experiments using the trilateration and EKF positioning algorithms at 40 different locations. At each location, we repeated the iteration of estimating the position 300 times. The results of the experiments are summarized in Figure 9. The X-axis represents the number of iterations and the Y-axis represents the errors of the positioning results in meters. A ‘+’ in the graph represents an error of trilateration positioning and is the average of the errors obtained at the 40 different locations. The dashed line (the points are so close that it appears to be a solid line) represents the errors of the EKF positioning and the straight line represents the average error of the trilateration (ET: from now on ET stands for “average error of the trilateration”) positioning. As the graph shows, the average of the errors of the trilateration is 4.07 m, whereas the error of EKF (EE: from now on EE stands for “average error of the EKF”) converges to 3.528m. There is a difference between ET and EE. This difference comes from the fact that ET only counts distances which are always positive. For example, consider the case where the estimates are (140, 200) and (160, 200). The trilateration would determine the user's position to be (150, 200) and ET would be 10. On the other hand, the estimate of EKF would be close to (150, 200) and EE would be close to 0.

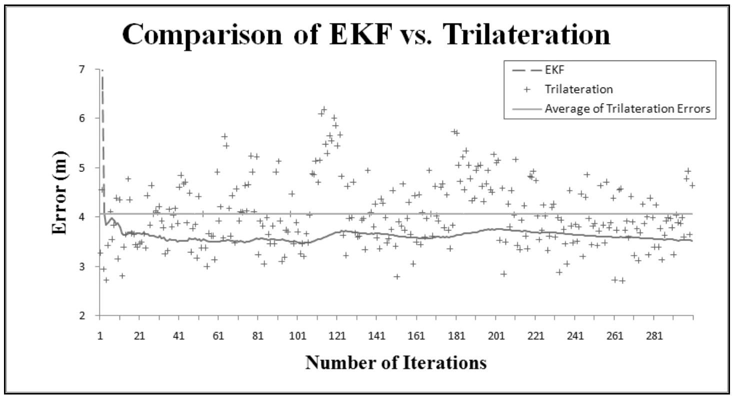


Figure 9. Comparison of Trilateration vs. EKF

3.2. Moving Object Database for ILBS

The major components of an indoor MODB system are the database server, the web server and the communication server. The communication server continuously receives (Time, Location) tuples from the mobile terminals. The Location is the coordinates of the measured location of the mobile terminal at the moment of Time. The communication server passes the received tuples to the database server. The web server delivers the user's commands to the database server and displays the answers from the database on the user's terminal. The database server stores all the (Time, Location) tuples from all the mobile terminals in the application field of the MODB. The database server also manipulates the queries from the web server and returns the answers of the queries to the web server referring to the digital map of the domain of the application.

[12] proposes an updating strategy with which we can save the communication cost without degrading the accuracy of database locations. The strategy applies the Kalman filter on the positions measured at $t_0, t_1, \ldots, t_i$ in order to estimate the state of the moving object and extrapolates the positions at the moments of $t_{i+1}, t_{i+2}, \ldots$, with the estimated state. If the difference between the extrapolated position and the measured position is not greater than the given threshold then the strategy omits sending the measured position. The strategy performs the extrapolation when the elements of P are all less than 0.001.

The algorithm of the strategy is shown in Table 4. For the first step, the algorithm applies the Kalman filter process on the consecutive i recently measured positions. We are using the decision tree method for the measurement. The formats of the matrices used in the Kalman filter process are as follows.

$$X_k = \begin{bmatrix} x_k \\ y_k \\ v_{xk} \\ v_{yk} \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The initial values for the other matrices are determined as follows.

$$Q=\begin{bmatrix}0.001 & 0 & 0 & 0\\ 0 & 0.001 & 0 & 0\\ 0 & 0 & 0.001 & 0\\ 0 & 0 & 0 & 0.001\end{bmatrix},\ R=\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix},$$

$$P_0=\begin{bmatrix}300 & 0 & 0 & 0\\ 0 & 300 & 0 & 0\\ 0 & 0 & 300 & 0\\ 0 & 0 & 0 & 300\end{bmatrix}.$$

After the first step, the algorithm checks the elements of P, and if they are all less than a small constant 'Epsilon' (0.001 for example) then the algorithm performs the extrapolation. In the process of the extrapolation, we replace all the Δts of A with 1s. Then, we multiply the A by the ith state obtained by the first step. The result of the multiplication is assigned to the variable 'prediction' in the algorithm. At this moment, if the difference between the i+1th measured location and the prediction is less than the threshold (3 meters, for example) then we skip sending the measured location to the communication server. Otherwise, we do not skip sending the measurement.

Table 4. The Algorithm for Updating

```
1) Apply the Kalman filter process shown in Figure 1 on the i measured locations to
   produce $\hat{x}_i$
2) measurement = i+1th measurement obtained by the positioning algorithm;
3) if (elements of $P$ < Epsilon) then {
4)      prediction = $A\hat{x}_i$;
5)      If (|prediction – measurement| < Threshold)
6)          Then skip sending
7)          Else send measurement to the server
8) Else send measurement to the server: }
```

If we skipped sending the i+1th measured location then we keep applying the updating algorithm on the i+2th measurement, i+3th measurement, and so on. When we apply the algorithm on the i+2th measurement, the Δts of A are all set to 2. When we apply the algorithm on the i+3th measurement, the Δts of A are all set to 3, and so on. However, the value of i does not change. In other words, we always apply the Kalman filter on the i recent measurements.

Sometime later, it happens that (|prediction – measurement| < Threshold) is no longer true. If (|prediction – measurement| < Threshold) is not true for n (n=3, for example) consecutive measurements, then we conclude that our prediction is no longer valid and let the mobile terminal send the n measurements to the server.

When the database server receives a query of "Where was the person A at the time xx?" it looks for (xx, Location) on the database. If it finds one then it returns the Location, otherwise it performs our Kalman filter based extrapolation and returns the result as the answer for the query. Since our updating policy skips updating when the difference between the prediction and the measurement is less than the Threshold, the answers returned by the database server are always within the Threshold from the measured location.

Our experimental results are summarized in Table 5. When we set Threshold to 3 meter, our strategy skipped sending 74.6 times in average. That is 52.5% of the measurements. Referring to Table 5, we can notice that the average error of our measured positions is 2.924. That is very close to our Threshold. That explains why the skip rate of our experiment is 52.5%. We can also notice that the average error (3.05214) of recorded positions for the experiment is a little higher than that (2.924) of "without updating policy" but they are very close. This is natural because we start submitting measurements when "(|prediction – measurement| < Threshold)" holds.

In the other set of experiments, we set Threshold to 2 meter. Our strategy only skipped sending 37 times, or 26%, in average. It is natural that the smaller value for the Threshold results in the smaller number of skips. As the number of skips gets smaller, the average of the errors of the recorded measurements for this experiment gets closer to that of "Without Updating Policy."

With these experimental results, we can conclude that our updating method can save about a half of the communication cost without almost any tradeoffs for indoor moving objects databases when the moving objects measure their positions with the WLAN based K-NN positioning method.

Table 5. Summary of Experiments

	Without Updating Policy	Threshold = 3 meter	Threshold = 2 meter
Number of skips	0	74.6 (52.5%)	37 (26%)
Average of errors (meter)	2.924	3.05214	2.956

We expected that our updating algorithm would save the communication cost by more than 70% in the test bed because the track consisted of only three straight lines. Saving only 52.5% is not satisfactory. The reason of this disappointing test results stems from the assumption that the Kalman filter makes. The Kalman filter assumes that the measurements are normal distributed around the real value. This implies that if we perform our positioning algorithm many (100) times at a point, P, without moving around, then the average of the measured positions should be P. In the reality, the average of the positions measured by our positioning algorithm does not exactly coincide with the real position.

3.3. Rendering Drawings

The purpose of the MODB is to answer spatio-temporal queries. Therefore, most answers are displayed on a map or a drawing for indoor applications. Our map rendering module displays a drawing recorded in an AutoCAD DXF (Drawing Interchange Format or Drawing Exchange Format) file and provides map manipulation functions such as zoom-in, zoom-out, and move. A DXF file consists of a large number of entities. The entities are classified into the arc, line, circle, and polyline types, among others. In DXF, an arc is determined by its attributes such as its center, radius, start angle, and end angle. The attributes of a line are the start and end points. Similarly, the attributes of each entity type is predefined in DXF. In DXF, every entity representation follows the entity type as shown in Table 6. The type of the entity shown in Table 6 is LINE, and the type name LINE follows "0," which indicates that the entity type appears in the next line. The "10" in the table indicates that the X coordinate of the start point appears in the next line. The others are obvious.

Table 6. A Part of the DXF File Used in the Proposed Rendering Module

code	Description
…	
0	Indicates the start of an entity
LINE	Entity type
…	
10	The next line is the X coordinate of the start point
35.88	X coordinate
20	The next line is the Y coordinate of the start point
13.65	Y coordinate
…	Z is omitted
11	The next line is the X coordinate of the end point
58.15	X coordinate
21	The next line is the Y coordinate of the end point
40.68	Y coordinate
…	A lot of entities and others

The main parts of our rendering module are the input, drawing, and manipulation parts. A flowchart of the input part is shown in Figure 9. The first method is PaperLoad( ), which uses the dialog tool of C# and returns the full path of the DXF file. It then continues reading two lines from the DXF file until the end of the file. We read two lines at a time because the entity type follows “0” in the DXF. The entity type determines the way of reading the attributes of the entity. Therefore, if the entity type (for example LINE) is read, we invoke the method (LineModule in this case) which reads in the attributes of the type. For each entity type, we define a class with local variables corresponding to the attributes of the entity. It also has methods including draw( ). For example, x_start, y_start, x_end, and y_end are some of the variables of the LINE class. For each entity read, our input module creates an object and appends it to the list called EntityList.

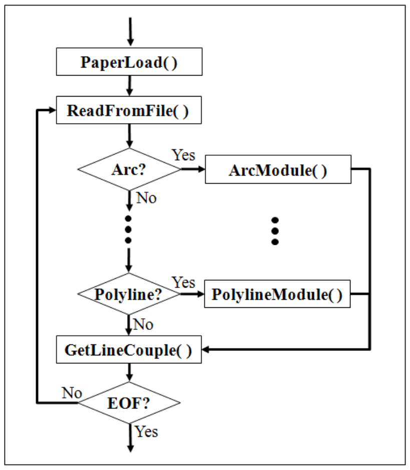


Figure 9. A Flowchart of the Input Part of our Rendering Module

Our drawing part draws all the objects listed in the EntityList. Before the actual drawing, it calculates the coordinates of the center of the drawing, paperMid and the center of the window, windowMid. The center of the drawing is defined by the following:

paperMidX = (XMin+XMax)/2,

paperMidY = (YMin+YMax)/2, (Expression 3)

where XMin is the minimum of x coordinates appearing in the entire DXF file. XMax, YMin, and YMax are all defined similarly. The center of the window is defined by the following:

windowMidX = drawPanelWidth / 2,

windowMidY = drawPanelHeight / 2, (Expression 4)

where drawPanelWidth (Height) is the width (height) of the panel where the drawing is drawn. We now calculate the scale for the drawing, which is basically the ratio of the window size to the drawing size. It is called mainScale and can be calculated by (Expression 5). We choose the minimum out of scaleX and scaleY and then multiply it by 0.9 so that the drawing can easily fit into the window.

mainScale = MIN(scaleX, scaleY) * 0.9, where scaleX = drawPanelWidth / (MaxX – MinX), scaleY = drawPanelHeight / (MaxY – MinY). (Expression 5)

Given a point (paperX, paperY) from the DXF file, we can calculate the point on the window, (winX, winY), by using (Expression 6). A point (paperX, paperY) in the DXF file is represented at (winX, winY) on the window. In (Expression 6), we use a single operand negation operation because Y coordinates increase from the top to the bottom for a window, whereas they increase from the bottom to the top for a drawing. Once we have the window coordinates, drawing an entity is just a matter of calling the right method provided by C#.

winX = (paperX – paperMidX) * mainScale + windowMidX,

winY = - (paperY – paperMidY) * mainScale + windowMidY. (Expression 6)

The manipulation part of the rendering module provides zoom in, zoom out, up, down, left, right, and so forth. It is easy to implement these functions. For example, zoom in (out) can be done by increasing (decreasing) the value of mainScale. For returning to the original scale, the OriginalScale( ) method is provided. Move up (down) can be done by increasing (decreasing) paperMidY, and move right (left) can be done by increasing (decreasing) paperMidX. Converting DXF coordinates into window coordinates can be done by performing (Expression 6). The inverse of (Expression 6) can be used for converting window coordinates into DXF coordinates.

Experimental results of the rendering program are shown in Figure 10. In the figure, screen captures after moving, zoom in, zoom out operations are included.

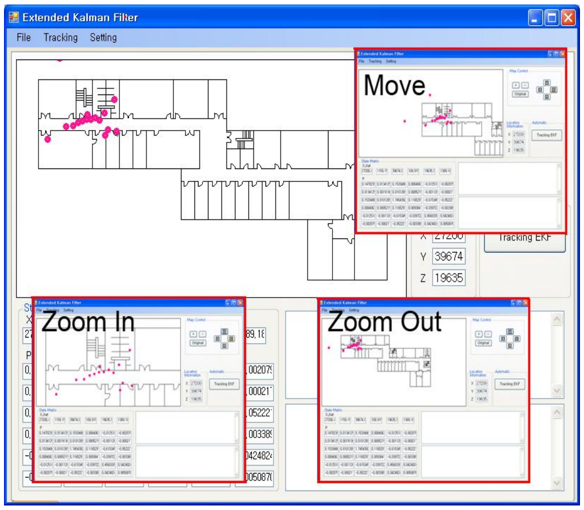
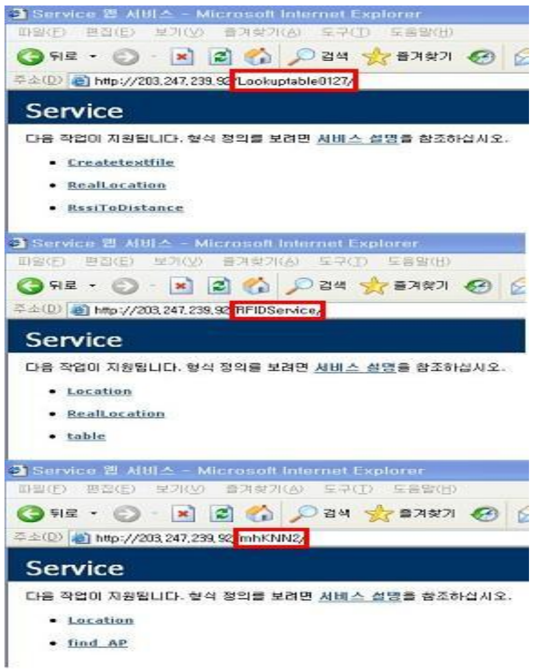


Figure 10. Experimental Results of the Rendering Module

3.4. Web Service

It is obvious that software reusing improves productivity a lot. Web service is one of the most convenient ways of reusing software. Therefore, my lab staffs have transformed those programs introduced in the previous subsections into web methods for the last year. Uploading location information of APs, returning the distance value corresponding to the input RSSI value, creating K-NN lookup tables, the real time phase of the K-NN, trilateration process, uploading RFID tag-ID with its location, and so on are few out of the implemented web methods. Implementation of these web methods is shown in Figure 11.


Service 웹 서비스 - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로
검색
즐겨찾기
주소(D)

http://203.247.239.92/Lookuptable0127/

Service

다음 작업이 지원됩니다. 형식 정의를 보려면 서비스 설명을 참조하십시오.

- Createtextfile
- RealLocation
- RssiToDistance

http://203.247.239.92/RFIDService/

Service

다음 작업이 지원됩니다. 형식 정의를 보려면 서비스 설명을 참조하십시오.

- Location
- RealLocation
- table

http://203.247.239.92/mhKNN2/

Service

다음 작업이 지원됩니다. 형식 정의를 보려면 서비스 설명을 참조하십시오.

- Location
- find_AP

Figure 11. Web Methods for Indoor Positioning

4. Conclusions

We have worked on indoor location based service for the last a few years and some of our research results are summarized in this paper. Now, we are planning to

1) Develop practical ILBS systems: integrating the above introduced methods.

2) Improve the methods introduced in this document: developing practical positioning techniques (for example, utilizing the sensors on the smart phones, figuring out more efficient algorithms), developing DB of drawings/contents, developing techniques to generate a drawing/contents by combining many small drawings/contents published on the Web, practical ontology, and so on.

References

[1] Mabrouk M, "OpenGIS Location Services (OpenLS): Core Services OGC 03-006r", http://www .opengis.org/

[2] Enge P, Misra P, "Special Issue on GPS: The Global Positioning System", Proceedings of the IEEE, **(1999)** January, pp. 3-172.

[3] Tekinay S, "Special Issue on Wireless Geolcation Systems and Services", IEEE Communications Magazine, **(1998)** April.

[4] Want R, Hopper A, Falcao V, Gibbons J, "The Active Badge Location System", ACM Transactions on Information Systems, vol. 10, no. 1, (1992) January, pp. 91-102.

[5] Harter A, Hopper A, "A New Location Technique for the Active Office", IEEE Personal Communications, vol. 4, no. 5, **(1997)** October, pp. 43-47.

[6] Priyanthat N, Chakraborty A, Balakrishnan H, "The Cricket Location-Support System", Proc. of 6th ACM International Conference on Mobile Computing and Networking, Boston, MA, **(2000)** August.

[7] Orr RJ, Abowd GD, "The Smart Floor: A Mechanism for Natural User Identification and Tracking", In Conference on Human Factors in Computing Systems (CHI2000) The Hague, Netherlands, **(2000)** April, pp. 1-6.

[8] Krumm J, et. al., "Multi-camera Multi-person Tracking for Easy Living", In 3rd IEEE Int'l Workshop on Visual Surveillance, Piscataway, NJ, **(2000)**, pp. 3-10.

[9] Bahl P, Padmanabhan V, "RADAR:An in-building RF-based user location and tracking system", INFOCOM 2000, **(2000)** March, pp. 775-784.

[10] Ladd AM, Bekris KE, Rudys A, Kavraki LE, Wallach DS, Marceau G, "Robotics-based Location Sensing Using Wireless Ethernet", Proceedings of the eighth Annual International Conference on Mobile Computing and Networking (MOBICOM-02), New York, **(2002)**, Sept. 23-28, pp. 227-238.

[11] Yim J, "Introducing a decision tree-based indoor positioning technique", Expert Systems with Applications, vol. 34, issue 2, **(2008)**, pp. 1296-1302.

[12] Yim J, Joo J, Park C, "A Kalman Filter Updating Method for the Indoor Moving Object Database", submitted for publication.

[13] Wolfson O, "Moving Objects Information Management: The Database Challenge", *Proceedings of the 5th Workshop on Next Generation Information Technologies and Systems* (NGITS'2002), Dan Caesarea Hotel, Caesarea, Israel, **(2002)** June 25-26, pp. 1-13.

[14] Olston C, Widom J, "Offering a Precision-Performance Tradeoff for Aggregation Queries over Replicated Data", 26th International Conference on Very Large Data Bases, Cario, Egypt, **(2000)** September.

[15] Tao Y, Xiao X, Cheng R, "Range Search on Multidimensional Uncertainty Data", *ACM Transactions on Database Systems*, vol. 32, Issue 3, **(2007)** August, pp. 1-54.

[16] Kilimci P, Kalipsiz O, "Moving Objects Databases In Space Applications", 3rd International Conference on Recent Advances in Space Technologies, **(2007)**, pp. 106-108.

[17] Ooi B, Huang Z, Lin D, Lu H, Xu L, "Adapting Relational Database Engine to Accommodate Moving Objects in SpADE", IEEE 23rd International Conference on Data Engineering, **(2007)**, pp. 1505 – 1506.

[18] Jin P, Wan S, Yue L, "Conceptual Modeling for Moving Objects Database Applications", 9th International Conference on Mobile Data Management, **(2008)**, pp. 217–218.

[19] Guting R, "How to Build Your Own Moving Objects Database System", International Conference on Mobile Data Management, **(2007)**, pp.1-2.

[20] Makki S, Sun B, Khojastehpour M, "An efficient information access scheme for mobile objects", IEEE International Conference on Information Reuse & Integration, **(2009)**, pp. 312-317.

[21] Han L, Wu J, Xie K, Ma X, Xu D, Zhang H, Chen Z, "A spatio-temporal database prototype for managing moving objects in GIS", IEEE International Geoscience and Remote Sensing Symposium.

[22] Zhang W, Li J, Zhang W, "Spatio-temporal Pattern Query Processing based on Effective Trajectory Splitting Models in Moving Object Database", First International Multi-Symposiums on Computer and Computational Sciences, **(2006)**, pp. 540-547.

[23] Kilimci P, Kalipsiz O, "Geometric Modeling and Visualization of Moving Objects on a Digital Map", Geometric Modeling and Imaging, **(2007)**, pp. 39-43.

[24] Kuijpers B, Vaisman A, "A Data Model for Moving Objects Supporting Aggregation", IEEE 23rd International Conference on Data Engineering Workshop, **(2007)**, pp. 546-554.

[25] Xiao Y, "Set Nearest Neighbor Query for Trajectory of Moving Objects", Sixth International Conference on Fuzzy Systems and Knowledge Discovery, **(2009)**, pp. 211-214.

[26] Liao W, Wu X, Zhang Q, Zhong Z, "Improving throughout of continuous k-nearest neighbor queries with multi-threaded techniques", IEEE International Conference on Intelligent Computing and Intelligent Systems, **(2009)**, pp. 438-442.

[27] Iijima Y, Ishikawa Y, "Finding Probabilistic Nearest Neighbors for Query Objects with Imprecise Locations", Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, **(2009)**, pp. 52-61.

[28] Ding H, Trajcevski G, Scheuermann P, "Efficient Similarity Join of Large Sets of Moving Object Trajectories", 15th International Symposium on Temporal Representation and Reasoning, **(2008)**, pp. 79-87.

[29] Schneider M, "Evaluation of spatio-temporal predicates on moving objects", 21st International Conference on Data Engineering, **(2005)**, pp. 516-517.

[30] Xiong X, Elmongui H, Chai X, Aref W, "Place: A Distributed Spatio-Temporal Data Stream Management System for Moving Objects", International Conference on Mobile Data Management, **(2007)**, pp. 44-51.

[31] Frihida A, Zheni D, Ghezala H, Claramunt C, "Modeling Trajectories: A Spatio-Temporal Data Type Approach", 20th International Workshop on Database and Expert Systems Application, **(2009)**, pp. 447-451.

[32] Ouri Wolfson A, Sistla P, Chamberlain S, Yesha Y, "Updating and Querying Databases that Track Mobile Units", invited paper, special issue of the *Distributed and Parallel Databases Journal (DAPD) on Mobile Data Management and Applications,* vol. 7, no. 3, **(1999)**, Kluwer Academic Publishers, pp. 257-288.

[33] Li X, Huang M, Bian F, "An Adaptive Dead-Reckoning Updating Policy for Continuous Query", International Conference on Artificial Intelligence and Computational Intelligence, **(2009)**, pp. 194-198.

[34] Jeung H, Liu Q, Shen H, Zhou X, "A Hybrid Prediction Model for Moving Objects", IEEE 24th International Conference on Data Engineering, **(2008)**, pp. 70-79.

[35] Wang X, Wang S, Wang Z, Lv T, Zhang X, "A New Position Updating Algorithm for Moving Objects", First International Multi-Symposiums on Computer and Computational Sciences, **(2006)**, pp. 496-503.

[36] Yu X, Chen Y, Rao F, Liu D, "Filtering location stream in moving object database", 15th International Workshop on Database and Expert Systems Applications, **(2004)**, pp. 645-649.

[37] http://code.google.com/intl/ko/apis/maps/documentation/javascript/

[38] Bruntrup R, Edelkamp S, Jabbar S, Scholz B, "Incremental map generation with GPS traces", Proc. of IEEE Intelligent Transportation Systems, **(2005)**, Sept. 13-15, pp. 574-579.

Authors

Jaegeol Yim received the M.S. and Ph.D. degrees in Computer Science from the University of Illinois at Chicago, in 1987 and 1990, respectively. He is a Professor in the Department of Computer Science at Dongguk University at Gyeongju Korea. His current research interests include Petri net theory and its applications, Location Based Service, AI systems, and multimedia systems. He has published more than 50 journal papers, 100 conference papers (mostly written in Korean Language), and several undergraduate textbooks.

Gyeyoung Lee received the M.S. degree in Computer Science from the Dongguk University in 1982 and the Ph.D. degree in Computer Engineering from the Dankook University in Korea in 1992, respectively. He is a Professor in the Department of Computer Science at Dongguk University at Gyeongju, Korea. His current research interests include Petri net theory, AI systems and Speech processing systems.