Kalman Filter vs. Particle Filter in Improving K-NN Indoor Positioning

Jaegeol Yim[1], Jinseog Kim[2], Gyeyoung Lee[1], and Kyubark Shim[2]

[1] Dept. of Computer Engineering, Dongguk University
Gyeongju, Korea
{yim,lky}@dongguk.ac.kr
[2] Dept. of Statistics and Information Science, Dongguk University
Gyeongju, Korea
jinseog.kim@gmail.com, shim@dongguk.ac.kr

Abstract. The Kalman filter has been widely used in estimating the state of a process and it is well known that no other algorithm can out-perform it if the assumptions of the Kalman filter hold. For a non-Gaussian estimation problem, both the extended Kalman filter and particle filter have been widely used. However, no one has performed comparison test of them. In the consequence, they arbitrarily choose one of them and apply it on their estimation process. Therefore, we have compared the performance of the Kalman filter against the performance of the particle filter. One of the practical fields on which these filters have been applied is indoor positioning. As the techniques of manufacturing mobile terminals have made a big progress, the demand for LBS (location based services) also has rapidly grown. One of the key techniques for LBS is positioning, or determining the location of the mobile terminal. Outdoor positioning is not a big burden to system developers because GPS (Global Positioning System) provides pretty accurate location information of a mobile terminal if the line of sight is not blocked. On the contrary, there is no practical solution for the indoor positioning problem. We can obtain exact location of a mobile terminal if we invest large amount of money, but this is economically not practical. One of the most practical candidate solutions for the indoor positioning problem is the WLAN (Wireless Local Area Network) based positioning methods because they do not require any special devices dedicated for indoor positioning. One of the most significant shortcomings of them is inaccuracy due to the noise on measured data. In order to improve the accuracy of WLAN based indoor positioning, both the Kalman filter and the particle filter processes have been applied on the measurements. This paper introduces our experimental results of comparing the Kalman filter and the particle filter processes in improving the accuracy of WLAN based indoor positioning so that indoor LBS developers can choose appropriate one for their applications.

Keywords: Indoor Positioning, K-NN, Kalman filter, Particle filter, fingerprinting method.

1 Introduction

A location based service (LBS) provides very useful services such as navigation, troops control, logistics, and so on to the users based on the geographic locations of the users and items. Therefore, positioning users and items is the most essential technique in development of a location-based service system [1]. There are numerous 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].

LBS are so useful that providing indoor LBS is very desirable. For instance, many huge buildings in metropolitan area, large scale companies, factories, universities, huge (underground) shopping malls, and so on are especially demanding LBS. Among the existing positioning systems, wireless local area network (WLAN) based positioning techniques are most interesting in the field of indoor LBS because of the following reasons. GPS signal is not available inside of a building so GPS system cannot be an indoor positioning and the others require special equipments dedicated for positioning. On the other hand, WLAN positioning systems do not require additional hardware dedicated for positioning and wireless network is being serviced everywhere including college campuses, airports, hotels and homes, so it is more economical and less time consuming than other indoor positioning techniques.

A WLAN-based positioning system determines a user's position referring to the received signal strengths (RSS) of the signals from access points (APs). However, it has a serious shortcoming. That is, RSS is influenced by so many environmental parameters and even K-NN (Nearest Neighbor) method which is the most accurate method of RSS based positioning is not accurate enough to be used in a practical application system.

Kalman filter and particle filter are good candidate tools in dealing with such noisy data as RSS. In fact, [11,12] has already introduced extended Kalman filter method for indoor positioning and [13] has already applied the particle filter to improve WLAN based indoor positioning. The process of Kalman filter iteratively predicts and corrects the prediction with measurements until some termination criteria met. The particle filter models the hidden state as a spatial posterior probability density function (PDF). Then, similarly to the Kalman filter, a particle filter also iterates prediction and correction, but it predicts and corrects the PDF not the state itself.

It is well known that the Kalman filter yields the best result if the assumptions (independent, white, normal distribution, and so on) of the Kalman filter hold. However, no one knows which one is better for a non-Gaussian estimation problem. If we know which one performs better than the other then we would not bother implementing both filters in developing an indoor LBS. Therefore, this paper introduces our experimental results of comparing the Kalman filter and the particle filter processes in improving the accuracy of K-NN WLAN based indoor positioning.

2 Related Works

We are comparing the Kalman filter against the particle filter in improving K-NN WLAN based indoor positioning. Therefore, K-NN WLAN based indoor positioning, the Kalman filter and the particle filter are the subjects related to our work.

2.1 K-NN WLAN Based Indoor Positioning

In the K-NN process, 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 the candidate points we measure the RSSIs many times. Let RSSIij denote the j-th received signal strength of the signal sent by APi. A row of the look-up table is an ordered pair of (coordinate, a list of RSSIs). 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, RSSI1, RSSI2, ..., where RSSIi is an average of signal strengths RSSIij received at (x, y) and sent by APi. An example of look-up table is shown in Table 1.

In the second phase, or real-time phase, the positioning program gathers RSSIs the user receives at the moment. If the positioning program is running on the user's handheld terminal, then the terminal itself will collect RSSIs. Let $X = (RSSI1, RSSI2, \ldots)$ be the vector of the collected RSSIs, K-NN then searches the look-up table to find the K closest candidate points and returns the average of the K coordinates 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 the coordinates of i-th C.P, APi is the MAC address of i-th 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
...	...	...	...	...	...

2.2 Kalman Filter

There are hundreds of papers on the Kalman filter, most of which involve its application to autonomous or assisted navigation [14,15,16,17], whereas there have been a few reports on its application to indoor positioning or navigation [18]. The Kalman filter process is summarized in Fig. 1. A state of a moving object is represented as x in the process. The vector x consists of the moving object's x, y coordinates and the velocity as shown in Fig. 2. The matrix A is to project the next state from the current state and the value of A is shown in Fig. 2 where in A represents the time elapsed from the current to the next states. The measurement is represented as the vector z consisting of x and y coordinates. In order to select x-y coordinates from x, H should be as shown in Fig. 2. The Kalman process iteratively predicts and corrects the prediction with measurement. During the prediction, it projects the next state and the

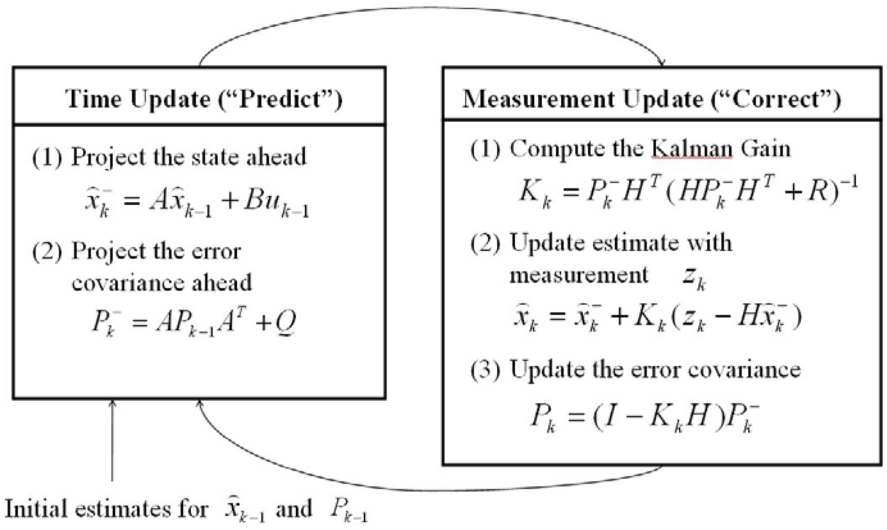
Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman Gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Fig. 1. The Kalman filter process

$$\hat{x}_k = \begin{pmatrix} x_k \\ y_k \\ v_{xk} \\ v_{yk} \end{pmatrix}, , A = \begin{pmatrix} 1 & 0 & \triangle t & 0 \\ 0 & 1 & 0 & \triangle t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, H = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$$

Fig. 2. The matrices used in the Kalman filter

error covariance, P. During the correction, it computes the Kalman gain, update estimate with measurement, and update the error covariance.

2.3 Particle Filter

The particle filter process is also a repetitive process. After initialization, it repeats importance sampling step and resampling step as shown in Fig. 3[19].

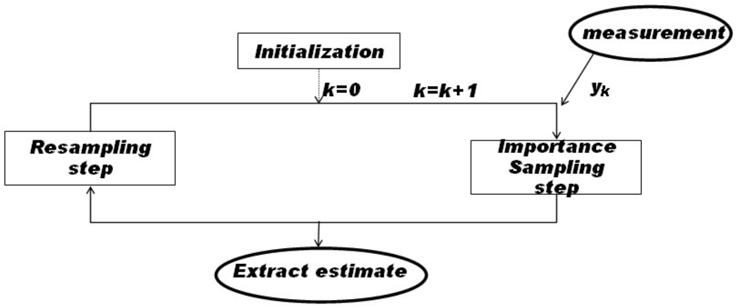

Fig. 3. A schematic diagram of the particle filter process

During the initialization, it populates particles, or samples, in the probability distribution function of the initial state of the dynamic system we are concerned. During the importance sampling step, considering the measurement at the moment, it assigns weights to the particles. Then, it determines the state at the moment as the weighted average of the particles. During the resampling step, it generates the next generation of particles.

3 Implementation of Kalman Filter

We have implemented our Kalman filter process as shown in Table 2 in C# using Microsoft visual studio. We initialize matrices Q, R, P, and X as shown in Table 4. Q and R are supposed to be the 'process noise covariance' and 'the measurement noise covariance' matrices respectively, but they are unknown. Besides, we know that it is the ratio of Q over R that really affects the performance of the Kalman filter [18]. Since the measurement noise is so big, we initialized R with much bigger number than Q's elements. X in the Kalman filter is corresponding to the state of the process which is the location of the mobile terminal in our experiment. Therefore, we initialize X with the first measurement. The matrix P represents the estimate error matrix. Since we already know our measurement error is so big, we initialize P with a big number. R and Q are the design parameters of the process and we have executed the process on the K-NN results shown in Table 1 with various values for R and Q. After many experimental executions, we have found the appropriate values for them as shown in Table 4.

Table 2. Our Kalman Filter Process

```
1. Initialize:
   For ( $i = 2$; not EOF; $i++$) {
2. Step (1) of Time Update
3. Step (2) of Time Update
4. Step (1) of Measurement Update
5. Step (2) of Measurement Update. Use the $i$-th measured location as $z$. Print $\hat{x}_k$.
6. Step (3) of Measurement Update
   }
```

$$Q = \begin{pmatrix} 0.001 & 0 & 0 & 0 \\ 0 & 0.001 & 0 & 0 \\ 0 & 0 & 0.001 & 0 \\ 0 & 0 & 0 & 0.001 \end{pmatrix}, \quad R = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad P_0 = \begin{pmatrix} 300 & 0 & 0 & 0 \\ 0 & 300 & 0 & 0 \\ 0 & 0 & 300 & 0 \\ 0 & 0 & 0 & 300 \end{pmatrix}, \quad \hat{x}_k = \begin{pmatrix} 85175.4 \\ 12225.5 \\ 0 \\ 0 \end{pmatrix}$$

Fig. 4. Initial values of our Kalman filter process

4 Implementation of the Particle Filter

We have implemented our Particle filter process as shown in Table 3 in C# using Microsoft visual studio. Assuming that the moving object moves along the x-axis at the speed of 788 mm/time unit, we initialize and to 0 and 788, respectively. We also assumed that the ranges of X and Y coordinates of the true positions are as follows:

$77290 \leq X \leq 81993$ and $9748 \leq Y \leq 17657$.

Therefore, we set the area for the initial particles as follows:

$74800 \leq X \leq 84490$ and $7500 \leq Y \leq 19850$.

According to [13], the appropriate number of particles is 400. So, we set the distance between a pair of adjacent particles parallel to the $X(Y)$ axis be 510(650) pixels for the initial particles.

At the moment, assuming that the mobile terminal can move to any direction, we use "RandomNormal$(0, 1.0\pi$);", and that the speed can also change dramatically, we use "RandomNormal(788, 700);" in the procedure of generating particles, where RandomNormal returns a random number in Normal(mean,

Table 3. Our particle filter process

```
1. Initialization. Generate initial particles (1 particle/$m^2$). For each particle, let 1 be
   its weight.
                    $\theta = 0$ (in rad), $V = 788 (mm/s)$
2. for ( $i = 1$; not EOF; $i++$) {
   (a) $z_i$ = the $i$-th measured location;
   (b) for each particle $x_i$ , compute its weight $w_i$ as follows:
         $w_i = w_i p[z_i|x_i]$, where $p[z_i|x_i] = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(z_i - x_i)^2}{2\sigma^2}\right)$
       where $\sigma = 20,000$.
   (c) TotalOfWeight $= \sum_{j=1}^{n} w_j$ , where $n$ is the number of particles
   (d) Normalize the weights so that the total of them is 1.
   (e) Print the weighted sum, $(X, Y, \theta, V)$, of all the particles
   (f) Particle = $Propagation(X, Y, \theta, V)$
   }
   Propagation($X, Y, \theta, V, TotalOfWeight$){
      if ($TotalOfWeight < 0.000001$)
         return(1 particle/$m^2$); // For each particle, let 1 be its weight
      else { // generate numberOfParticles (400, for example) particles
         for (i=0; I ¡ numberOfParticles; i++) {
            $Temp\theta = +RandomNormal(0, 1.0\pi)$;
            $TempV = V + RanddomNormal(0.0, 300)$;
            Particle[i].X = X + TempV*cos(Temp$\theta$ );
            Particle[i].Y = Y + TempV*sin(Temp$\theta$ );
            Particle[i].$\theta$ = Temp$\theta$ ;
            Particle[i].V = TempV;
         } // end of for
         return(Particle);
      } // end of else
   }
```

standard deviation) distribution. Now, the remaining parameter is the weight function for step 2.(b) in Table 3. As the weight of a particle, we use the inverse of the distance between the particle and the posterior generated in the previous iteration. The following is the actual sentence used in our program:

```
particles[j].weight
    = 1/(Math.Sqrt((x - particles[j].X) * (x - particles[j].X)
      + (y - particles[j].Y) * (y - particles[j].Y)));
```

5 Experiments

We are comparing the Kalman filter and the particle filter processes in improving the accuracy of WLAN based indoor positioning. Therefore, we have performed the K-NN positioning while we are walking through the path shown in Fig. 5. We have walked through the path 4 times. A part of our test results is shown in Table 4. The unit of the measurements is AutoCAD unit. In our experiments,

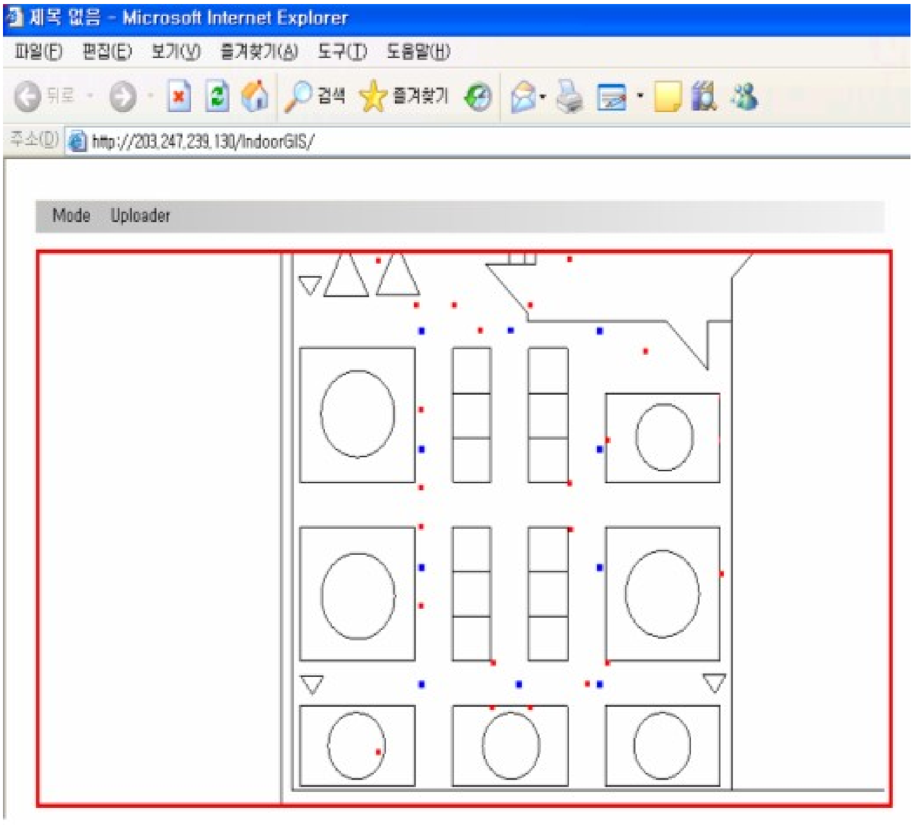
제목 없음 - Microsoft Internet Explorer
파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)
뒤로
검색
즐겨찾기
주소(D) http://203.247.239.130/IndoorGIS/
Mode Uploader

Fig. 5. A typical test result of K-NN indoor positioning

Table 4. A part of the 1-NN results (average error is 6,240.03 and standard deviation is 2,732.15)

	True X	True Y	Measured X	Measured Y	Error
Point 1	79,626	17,657	85,175	12,225	7,765
Point 2	80,414	17,657	77,187	7,444	10,711
Point 3	81,203	17,657	82,190	10,207	7,515
...	...	...	...	...	...
Point 29	78,069	17,657	76,187	19,217	2,444
Point 30	78,847	17,657	79,172	18,222	651

0.5 meter is equivalent to about 1,000 units. That is, 6420, the average error of K-NN obtained by the experiment, is equivalent to about 3.21 meters.

Then we have run the Kalman filter process on the data shown in Table 4 with various R and Q. Our test results are summarized in Table 5. R and Q matrices represent measurement noise covariance and process noise covariance, respectively. We found that our measurement noise covariance is 7025691.4 from Table 4. However, process noise covariance is hard to obtain. After many experiments, we have found that the accuracy of the Kalman filter process depends on the ratio of R value over Q value. Therefore, we have fixed R to 1 and changed the value of Q. After many experiments, we have found that $R = 1$ and $Q = 0.0001$ is appropriate. We have walked around the path on Fig. 5 4 times. The row labeled "For all" of Table 5 represents the average of all the 4 laps while "first lap" represents the average of the first lap. In summary, our Kalman filter process improves the accuracy of our K-NN process by 6421/3323 = 1.93 times. In other words, the error is reduced by almost half if we use the Kalman filter process. The unit of the figures shown in Table 5 is AutoCAD pixel and 3323 is equivalent to about 3.323 meters.

Table 5. A summary of our Kalman filter test results

R	1	1	1	1	1	1	7025691.4	7025691.4
Q	0.1	0.01	0.001	0.0001	0.000055	0.00001	700	1.0
For all	4975	4233	3459	3323	3355	3435	3376	4071
first lap	4855	4304	4207	4160	4156	4152	4028	5883

In the first experiment of running our particle filter, we have tried the following two weight functions: 1/(distance**2) and 1/(distance**4). Our test results are summarized in Table 6. The test results suggest 1/(distance**2) as the weight function. Considering the average error of K-NN, 6421, the particle filter improves the accuracy by 14% if 1/(distance**2) is used as the weight function. However, considering the average error of the Kalman filter, 3323, there should be a room to improve the particle filter process.

Table 6. Test results for simple weight functions

Weight	K-NN	1/distance	1/(distance**2)	1/(distance**4)
Average error	6240	5854	5534	12415

Therefore, we have tried the weight function shown in Table 3 with various values for σ, the standard deviations of RandomNormal, and found that the values shown in Table 3 are appropriate and our test results are summarized in Table 7.

Table 7. Test results for another simple weight functions

Weight	K-NN	$\sigma = 12,000$	$\sigma = 20,000$	$\sigma = 30,000$
Average error	6,240	3,716	3,648	3,703

Chao et al.[13] assigned the weight of 0 to the particles which are located at invalid area, for example, located beyond a wall. With this strategy, we could improve our particle filter process a little further as shown in Table 8.

Table 8. Test results for location-constrained weight function

Weight	K-NN	Kalman filter	Particle filter $\sigma = 20,000$	Location-constrained weight
Average error	6,240	3,323	3,648	3,572.4

Chao et al.[13] also suggested the strategy of removing a particle if its location is invalid during the Propagation. We have tried this strategy with various sizes of the valid area. However, this strategy did not improve the accuracy of our particle filter as shown in Table 9.

Table 9. Test results for location-constrained particle generation

Weight	K-NN	Kalman filter	Particle filter $\sigma = 20,000$	Size=3m	Size=2m	Size=0.6m
Average error	6,240	3,323	3,648	3,980	4,000	4,052

6 Conclusions

Chao et al. [13] and Yim et al.[18] introduced particle filter application and Kalman filter application on K-NN WLAN based indoor positioning. We have compared these two methods and found out that their accuracies are almost same. Chao et al. showed that the accuracy of the particle filter can be almost doubled if we make use of location-constraint. We have tried the location-constrained weight strategy and found out it can improve the accuracy of our particle filter process a little. We have also tried the location-constrained particle generation strategy. However, it did not contribute any in improving our particle filter process. Instead, as we decreased the size of the valid area, the running time of our process significantly increased. This implies that the performance of the particle filter changes drastically as the characteristics of measured data change.

The particle filter takes a long time for populating the particles. However, the process is intuitive and easy to implement. It is also easy to modify an implemented particle filter to adapt it to the new set of data. On the other hand, the Kalman filter is easy to tune R and Q to get the better results. However, it is not so easy to modify an implemented Kalman filter to adapt it to a different kind of measurements. In the case of Kalman filter, if we assign small numbers to the elements of R, it is guaranteed for the Kalman filter to yield at least the measurements. On the other hand, in the case of particle filter, the result can be worse than the measurements if we use wrong formula for weight calculation or population generation. In conclusion, if both are tuned to get their best results, then their performance is almost same and each of them has its own strength and weakness. Therefore, in practice, we should choose appropriate one considering the strength and weakness, characteristics of measured data and the situation of application.

Acknowledgments. This research was supported by Basic Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2011-0006942).

References

1. Yim, J.: Introducing a decision tree-based indoor positioning technique. Expert Systems with Applications 34(2), 1296–1302 (2008)
2. Enge, P., Misra, P.: The Global Positioning System. Proceedings of the IEEE Special Issue on GPS, 3–172 (1999)
3. Tekinay, S.: Special Issue on Wireless Geolocation Systems and Services. IEEE Communications Magazine (April 1998)
4. Want, R., Hopper, A., Falcao, V., Gibbons, J.: The Active Badge Location System. ACM Transactions on Information Systems 10(1), 91–102 (1992)
5. Harter, A., Hopper, A.: A New Location Technique for the Active Office. IEEE Personal Communications 4(5), 43–47 (1997)
6. Priyanthat, N., Chakraborty, A., Balakrishnan, H.: The Cricket Location-Support System. In: Proc. of 6th ACM International Conference on Mobile Computing and Networking, Boston, MA (August 2000)
7. Orr, R.J., Abowd, G.D.: The Smart Floor: A Mechanism for Natural User Identification and Tracking. In: Proceedings of the Conference on Human Factors in Computing Systems (CHI 2000), The Hague, Netherlands, pp. 1–6 (April 2000)
8. Krumm, J., et al.: Multi-camera Multi-person Tracking for Easy Living. In: Proceedings of the 3rd IEEE Int'l Workshop on Visual Surveillance, Piscataway, NJ, pp. 3–10 (2000)

9. Bahl, P., Padmanabhan, V.: RADAR: An in-building RF-based user location and tracking system. In: Proceeding of INFOCOM 2000, pp. 775–784 (March 2000)
10. Ladd, A.M., Bekris, K.E., Rudys, A., Kavraki, L.E., Wallach, D.S., Marceau, G.: Robotics-based Location Sensing Using Wireless Ethernet. In: Proceedings of the Eighth Annual International Conference on Mobile Computing and Networking (MOBICOM 2002), New York, pp. 227–238 (2002)
11. Kotanen, A., Hannikainen, M., Leppakoski, H., Hamalainen, T.D.: Experiments on local positioning with bluetooth. In: Proceedings of International Conference on Information Technology: Coding and Computing (Computers and Communications) (ITCC 2003), pp. 297–303 (April 2003)
12. Qasem, H., Reindl, L.: Unscented and Extended Kalman Estimators for non Linear Indoor Tracking Using Distance Measurements. In: Proc. of the 4th Workshop on Positioning, Navigation and Communication, WPNC 2007, pp. 177–181 (March 2007)
13. Chao, C., Chu, C., Wu, A.: Location-Constrained Particle Filter human positioning and tracking system. In: Proceedings of IEEE Workshop on Signal Processing System, pp. 73–76 (2008)
14. Teo, T., Chai, J., Yao, W.: Design of a positioning system for AGV navigation. In: Proc. of the 7th International Conference on Control, Automation, Robotics and Vision (ICARCV 2002), pp. 637–642 (2002)
15. Peroutka, Z.: Design considerations for sensorless control of PMSM drive based on extended Kalman filter. In: Proc. of 2005 European Conference on Power Electronics and Applications, pp. 1–10 (September 11-14, 2005)
16. Boussak, M.: Implementation and experimental investigation of sensorless speed control with initial rotor position estimation for interior permanent magnet synchronous motor drive. IEEE Transactions on Power Electronics 20(6), 1413–1422 (2005)
17. Bolognani, S., Tubiana, L., Zigliotto, M.: Extended Kalman filter tuning in sensorless PMSM drives. IEEE Transactions on Industry Applications 39(6), 1741–1747 (2003)
18. Yim, J., Park, C., Joo, J., Jeong, S.: Extended Kalman Filter for Wireless LAN Based Indoor Positioning. Decision Support Systems 45, 960–971 (2008)
19. Copsey, K.: Tutorial on Particle filters. Pattern and Information Processing Group DERA Malvern, K.Copsey@signal.dera.gov.uk