2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

# Accurate Mobile Robot Localization in indoor environments using Bluetooth

Aswin N Raghavan[1] Harini Ananthapadmanaban[2]
Manimaran S Sivamurugan[1] Balaraman Ravindran[3]
[1]Dept. of Computer Science and Engineering, National Institute of Technology, Tiruchirapalli
[2]Dept. of Electronics and Communication Engineering, National Institute of Technology, Tiruchirapalli
[3]Dept. of Computer Science and Engineering, Reconfigurable and Intelligent Systems Engineering Lab, Indian Institute of Technology Madras

*Abstract*— **In this paper, we describe an accurate method for localization of a mobile robot using bluetooth. We introduce novel approaches for obtaining distance estimates and trilateration that overcome the hitherto known limitations of using bluetooth for localization. Our approach is reliable and has the potential of being scaled to multi-agent scenarios. The proposed approach was tested on a mobile robot, and we present the experimental results. The error obtained was 0.427 $\pm$ 0.229 m, which proves the accuracy of our method.**

## I. Introduction

Localization is a fundamental problem in robotics. Location information is essential for planning and decision making processes. In this paper, we describe an accurate method for localization of a mobile robot using bluetooth. Bluetooth has several inherent advantages like low power consumption, an ubiquitous presence, low cost and easy availability. Furthermore, it is immune to electromagnetic chaos because of its Frequency Hopping Spread Spectrum(FHSS). It is easily scalable for multi-agent scenarios, wherein communication and localization go hand in hand. Other solutions like Wi-Fi, GPS(Global Positioning System), RFID etc. are not suitable for robot localization for the following reasons. GPS requires a Line-of-Sight with four satellites, thus not suitable for indoor environments. RFID is not capable of communication. Even though Wi-Fi has a higher data rate, its high power consumption makes it unsuitable. Other technologies such as Zigbee are not proliferated enough and not available in laptops, cell phones and PDAs.

A number of other bluetooth localization techniques have been presented till date. [1] reported that bluetooth can function very well as a localization tool. [3] concluded that bluetooth RSSI is not suitable for localization. [2] concluded that the inability of bluetooth to maintain connections makes it unsuitable. [4], [5], [6] and [7] achieved a mean error of 3.76 m,2.06 m, 1.52-3.0 m and 1.2 m respectively.

Our work is different from the above mentioned works in two ways:

1) Method of obtaining distance estimates: using a novel method of inquiry.
2) Trilateration: An novel trilateration method which improves accuracy.

The use of particle filters for localization is well-known. Apart from overcoming the limitations mentioned in the above citations, our work achieved a localization error of **0.427 $\pm$ 0.229 m**, proving that bluetooth is very much suitable for localization of mobile robots. A brief explanation about each step of the algorithm follows.

1) Obtaining signal strength indicator values from bluetooth beacons using our method of inquiry.
2) Converting the values got from the above step to an approximate distance. Since a one-to-one mapping is not possible, multiple distances are obtained.
3) Positions that satisfy the above distances with minimum error. Here, we introduce a new trilateration method. This step outputs multiple positions.
4) Using a particle filter to take each output of the previous step into account, making use of the motion model of the robot. Repeat from step1 again for the next move.

The following sections explain each step in detail. Section 2 explains relevant bluetooth specifications and the novel method of inquiry used. Section 3 describes a new trilateration technique. Section 4 explains about the localization algorithm with reference to particle filters. Experimental results and conclusions are presented in Sections 5 and 6.

## II. Bluetooth

Bluetooth is a short-range wireless technology that operates in the unlicensed 2.4GHz ISM band. Each bluetooth device has a unique 48-bit MAC address [8].

### A. Received Signal Strength Indicator(RSSI) and Power Control

Bluetooth devices are classified into 3 classes depending on their transmitting power level and range. In this work, we used off-the-shelf class-2 USB dongles with a range of $10m$ and maximum output power as $2.5mW/4dBm$. A bluetooth device maybe roughly divided into two parts: a controller(present in the dongle) and a host(present in the CPU). The controller consists of the bluetooth radio, the LMP(Link Manager Protocol) layer and a HCI(Host Controller Interface). LMP is used for link set-up and control.

HCI provides an interface to access the baseband controller and link manager.

Received Signal Strength Indicator(RSSI) is a parameter generated by the bluetooth radio. It is an indication of the power level of the received signal. Golden Receiver Power Range(GRPR) is a power level range defined by two threshold levels(upper and lower).

Bluetooth implements Adaptive Power Control i.e. the transmitted power is automatically increased or decreased if it differs too much from ideal characteristics, defined by the GRPR. Based on whether the RSSI is greater or lesser than the GRPR, the transmission power level is either decreased or increased. The exact bounds of GRPR are not clearly defined and are manufacturer-dependent to minimize Bit-Error Rate([3] tried to guess the GRPR). Power control implementation is optional for Class-2 and Class-3 devices and mandatory for Class-1 devices.

Many works like [9] used the HCI command *hci_read_rssi* which requires a connection between the devices. It is difficult to establish and maintain multiple connections with a bluetooth device [2]. Moreover, *hci_get_rssi* does not return the RSSI value itself but the difference of RSSI and the limits of GRPR [8]. The above mentioned Adaptive Power Control takes place only after a connection is established. RSSI obtained using this method will vary over time due to adaptation, hence is not very informative. Another HCI command, *Inquiry_with_RSSI* exists, that returns the actual RSSI value, at the time of inquiry i.e. without making any connections. [6] mentioned this but did not use it as their hardware did not support it. Since power control does not take place, the use of *Inquiry_with_RSSI* with *Inquiry_mode* set to $0x01$, makes RSSI more reliable and informative. Bluetooth devices have a limit on the number of connections that can be maintained at a given time(usually 7). Since no connections are made in our method, this drawback is eliminated.

The required commands were sent to the controller using BlueZ/C and the events and functions defined therein. It involved the following steps: 1. Set Inquiry mode to 0x01. 2. Configure parameters for *Inquiry_with_rssi* viz. Duration of inquiry, number of responses, discovery mode of devices to discover(GIAC or LIAC). 3. Send the command *inquiry_with_rssi* to the controller. 4. Wait for event *inquiry_result_with_rssi* to occur. 5. Extract RSSI values from the packet returned. 6. Repeat till event *inquiry_complete* occurs. A timeout of 3s was used.

### B. Variation of RSSI with distance

As distance between two bluetooth devices increases, the RSSI value is expected to fall. The aim is to obtain a mapping from RSSI to distance. However, due to effects of interference and multipathing, a one-to-one mapping is not possible. [1] notes that RSSI varies even for a stationary object(Fig. 1 and 2). [10] cites three methods to obtain this mapping. In this work, interpolation along with motion is used to obtain the mapping. The robot executes straight-line motion in steps, stopping after every step to perform an inquiry. The variation of RSSI as the robot moves away from a beacon is shown below(Fig. 3). This method of mapping RSSI to distance requires significantly lesser memory and training time as compared to fingerprinting. During localization, an observed RSSI vector from 3 beacons, say $(r_1, r_2, r_3)$ corresponds to many distance triplets $(d_1, d_2, d_3)$, each of which is considered by the trilateration algorithm and then by the particle filter.

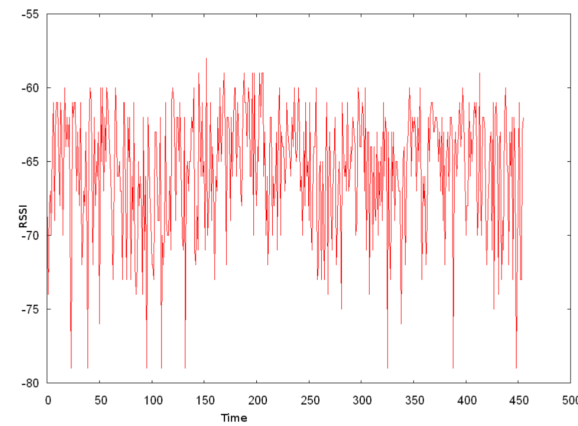


Fig. 1: Variation of RSSI v/s Time at a fixed distance of 3m

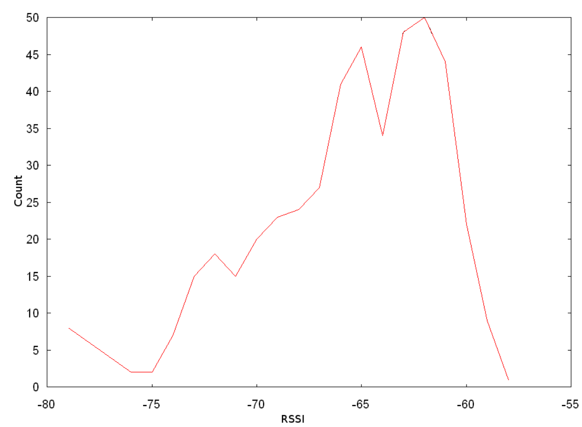


Fig. 2: Histogram of the graph shown in Fig. 1

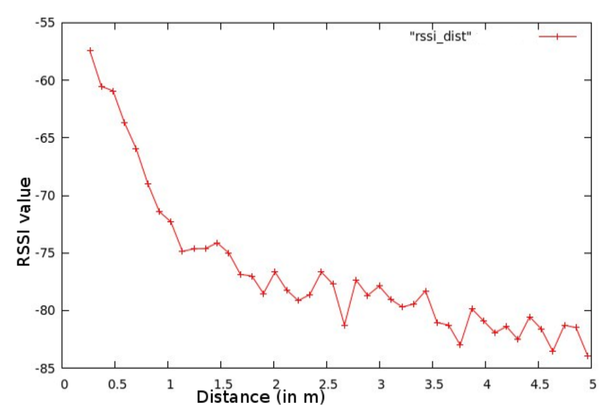


Fig. 3: Observed RSSI variation as the robot moves away from the beacon at a constant velocity

## III. TRILATERATION

Trilateration is a method to determine the position of an object based on simultaneous range(distance) measurements from three or more reference points at known locations. Considering the ideal case, the 3 circles obtained from the reference centers $(x_i, y_i)$ and distances $r_i$, will intersect at exactly one point. But in the case of noisy measurements, the circles may intersect in an area or not intersect at all. In such cases, the solution which gives the minimum error must be considered. Since small $r_i$ are more accurate, our method minimizes the total relative error instead of the absolute error. This is explained in detail below.

### A. Over-estimated system of equations

Now consider a system of $n$ linear equations in $m$ variables. This can be written as $AX = B$ where $A$ is $n$ X $m$ coefficient matrix, $X$ is a $m$ X 1 variable matrix and $B$ is the $n$ X 1 constant matrix. We can write the solution as $X = A^{-1}B$ only if $n = m$ and $A^{-1}$ exists. If $n > m$, we have an over determined system of equations which can be solved using the pseudo inverse method. $X = A^{+}B$ where, $A^{+}$ is the *pseudo-inverse* of $A$. On solving, this will give the least square error of the solution when $n > m$. So the system of circle equations is represented as

$$X = \begin{pmatrix} s \\ x \\ y \end{pmatrix}$$

$$A = \begin{pmatrix} 1 & -2x_1 & -2y_1 \\ 1 & -2x_2 & -2y_2 \\ . & . & . \\ . & . & . \\ 1 & -2x_n & -2y_n \end{pmatrix}$$

$$B = \begin{pmatrix} {r_1}^2 - {x_1}^2 - {y_1}^2 \\ {r_2}^2 - {x_2}^2 - {y_2}^2 \\ . \\ . \\ {r_n}^2 - {x_n}^2 - {y_n}^2 \end{pmatrix}$$

where, $s = x^2 + y^2$. But the problem with this method was that it is based on absolute error and hence not accurate. The solution obtained from this method was used as the initial estimate for the iterative trilateration described below.

### B. Iterative Trilateration

The method used here is an iterative algorithm based on gradient descent to find the point of least error. Such a method was used by [11]. Let the reference points and the corresponding distances be denoted by $(x_i, y_i)$ and $d_i$ respectively. A trivial initial estimate is considered $(x_e, y_e)$. The difference or error in the estimated distance and the measured distance is calculated as

$$|f_i| = \left| d_i - \sqrt{(x_i - x_e)^2 + (y_i - y_e)^2} \right|$$

Now applying the first degree Taylor series approximation, the adjustment $(\Delta x, \Delta y)$ used in the iteration of $(x_e, y_e)$ can be determined using matrix calculation with the following equations

$$\Delta = (B^T B)^{-1} B^T f \quad or \quad \Delta = \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

where B is given by

$$B = \begin{pmatrix} \frac{\partial f_1}{\partial x_e} & \frac{\partial f_1}{\partial y_e} \\ \frac{\partial f_2}{\partial x_e} & \frac{\partial f_2}{\partial y_e} \\ . & . \\ . & . \\ \frac{\partial f_i}{\partial x_e} & \frac{\partial f_i}{\partial y_e} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{(x_1 - x_e)}{\sqrt{(x_1-x_e)^2+(y_1-y_e)^2}} & \frac{(y_1 - y_e)}{\sqrt{(x_1-x_e)^2+(y_1-y_e)^2}} \\ . & . \\ . & . \\ \frac{(x_i - x_e)}{\sqrt{(x_i-x_e)^2+(y_i-y_e)^2}} & \frac{(y_i - y_e)}{\sqrt{(x_i-x_e)^2+(y_i-y_e)^2}} \end{pmatrix}$$

The update equation is,

$$x_e = x_e + 0.05\Delta x$$

$$y_e = y_e + 0.05\Delta y$$

The step size was reduced to 0.05 times $\Delta$, since, otherwise it was too large for convergence. Moreover, the error function $f_i$ calculates the absolute error. This biased the solution towards larger circles as they may have larger absolute error. Considering the fractional error will lead to a more correct point of MMSE as this gives equal weightage to different circles. This is important as smaller distances are more accurate. So $f_i$ is modified as

$$|f_i| = \left| \frac{d_i - \sqrt{(x_i - x_e)^2 + (y_i - y_e)^2}}{d_i} \right|$$

to represent the **fractional error**. The iterations are performed until the error reduces to within 6 decimal places.

## IV. LOCALIZATION

*Localization* is the ability of a robot to locate itself within an environment. It is the estimation of the *pose* ie. position and orientation of the robot at any given instant of time. Localization is fundamental to truly autonomous robots, and enable them to execute many useful tasks such as office delivery, rescue operations etc. *Global localization* or *positioning* aims to determine the pose of the robot in a known environment like a learned map or the presence of landmarks or beacons. In this section, an *active global localization in a static environment using bluetooth* is described. The robot is aware of the location of the bluetooth beacons(landmarks). Refer [12] for more definitions.

### A. Problem Formulation

Localization is solved as an online filtering problem. Let $x_t$ denote the state(pose) of the robot at time $t$, $t = 0, 1, \dots k$. $x_t$ is the vector $[x\ y\ \theta]^T$, the position and orientation of the robot in the environment. Let $z_t$ denote the measurement vector at time $t$, $t = 1, 2, \dots k$. $z_t$ contains the trilaterated output as described in the previous sections viz. $[x\ y]^T$. Since off-the-shelf bluetooth dongles do not possess directional antennae, the angle $\theta$ does not appear in the measurement vector and in the measurement model. The problem of localization can be stated as computing the *posterior* density $p(x_k|z_{1:k})$. $p(x_0)$ is assumed as in initial distribution, in this case a uniform distribution over all possible locations $[x\ y]^T$.

### B. Bayesian Filtering

In theory, the posterior density can be computed recursively in two stages: *predict* and *update*. Suppose that $p(x_{k-1}|z_{1:k-1})$ is available as a prior PDF of $x_{k-1}$, prediction obtains the prior PDF of $x_k$ via the Chapman-Kolmogorov equation

$$p(x_k|z_{1:k-1}) = \int p(x_k|x_{k-1}, z_{1:k-1})p(x_{k-1}|z_{1:k-1})dx_{k-1}$$

If we assume a Markov process of order one, $p(x_k|x_{k-1}, z_{1:k-1}) = p(x_k|x_{k-1})$, which is the state transition probability. In the update stage, $z_k$ is used to update the predicted prior via Bayes rule

$$p(x_k|z_{1:k}) = \frac{1}{\eta}p(z_k|x_k)p(x_k|z_{1:k-1})$$

where the normalizing constant

$$\begin{aligned} \eta &= p(z_k|z_{1:k-1}) \\ &= \int p(z_k|x_k)p(x_k|z_{1:k-1}) \end{aligned}$$

depends on the likelihood $p(z_k|x_k)$ defined by the measurement model.

Bayesian filtering is optimal in the sense that it computes the posterior by using all the available information. Kalman Filter based approaches can be used as well. However, by using a Monte-Carlo sampling-based approach to localization, the following advantages are achieved [13]: 1. It can represent multi-modal distributions in contrast to the Kalman filter. 2. It drastically reduces the memory requirement as compared to Grid Based approaches. 3. It is more accurate than Markov Localization with a fixed cell size, since the discretization error is avoided. 4. It is easy to implement.

### C. Monte Carlo Localization

In Monte-Carlo Localization(*MCL*) based approaches, the required posterior $p(x_k|z_{1:k})$ at time $k$ is represented by a set of weighted samples

$$S_k = \{x^i, w^i\},\ i = 0, 1, 2 \ldots N_p$$

each containing a weight $w^i$, also called *importance* factor. The weights are normalized after each update so that $\sum_{i=1}^{N_p} w_i = 1$. To avoid intractable integration in the Bayesian statistics, the posterior density is represented by a weighted sum of these $N_p$ samples.

$$p(x_k|z_k) \approx \frac{1}{N_p}\sum \delta(x_k - {x_k}^i)$$

where $\delta$ is the Dirac delta function. For sufficiently large $N_p$, the summation approximates the true posterior density. In the particle filter implementation of MCL, the set of $N_p$ particles is recursively filtered in two stages: predict and update.

*1) Prediction:* In this step, the posterior $p(x_k|x_{k-1}, u_{k-1})$ at time $k$ is predicted from the belief state $p(x_{k-1}|z_{1:k-1})$ and a control vector $u_{k-1}$. The set of particles $S_{k-1}$ corresponds to the state $x_{k-1}$. The control action $u_{k-1}$ has to be applied to each particle in $S_{k-1}$ taking into account the motion model of the robot. This gives a sample set $S'_k = x'^i, w'^i, i = 0, 1, \ldots N_p$. Note that $w'^i = w^i$

*2) Update:* In this step, the measurement model is taken into account. Namely, each particle of $S'_k$ is weighted by the likelihood $p(z_k|{x_k}^i), i = 0, 1, ..N_p$. Now we have the new particle set $S_k$.

*3) Degeneracy:* A common problem with particle filters is degeneracy. After a few iterations, most of the particles have negligible weight and only few of $N_p$ particles contribute significantly to the posterior. This happens because most particles have drifted far away from the actual position and hence their weights (which is proportional to the likelihood of measurement) is negligible. Many *resampling* techniques have been suggested. In this work, the linear time resampling technique in [14], [15] has been used. To avoid the overhead of resampling at every iteration, the effective sample size(*ESS*) is computed. Only if the ESS drops below a certain threshold, resampling is performed. Resampling discards particles with negligible weight and duplicates the particles with considerable weight. ESS is computed as follows:

$$\begin{aligned} {cv_t}^2 &= \frac{var({w_t}^i)}{E^2({w_t}^i)} \\ &= \frac{1}{N_p}\sum_{i=1}^{N_p}(N_p w^i - 1)^2 \\ ESS_t &= \frac{N_p}{(1 + {cv_t}^2)} \end{aligned}$$

### D. Motion Model

This involves predicting the state of the particle, given its initial state and a control vector $u$. The state of the robot is represented by the vector $[x\ y\ \theta]^T$. A control vector is represented by $[d\ \theta_1\ \theta_2]^T$. That is, the robot executes $\theta_1$ units of rotation followed by $d$ units of translation and then $\theta_2$ units of rotation. Noise in translation and rotation, and drifting are considered. The mean and variance of error in translation, rotations and drift of the robot were calculated experimentally. This error was modeled as a *Gaussian* function. A random sample from this function is added as noise to the new predicted state. Let $N(\mu, \sigma, x)$ denote the normal function with mean $\mu$ and variance $\sigma$. For the Present state - $X = [x\ y\ \theta]^T$ and Control vector - $u = [\theta_1\ d\ \theta_2]^T$, the control vector with some added noise $u'$ is given by

$$\begin{aligned} u' &= [\theta'_1\ d'\ \theta'_2]^T \\ \theta'_1 &= \theta_1 + N(\mu_{rot}, \sigma_{rot}, RANDOM) \\ \theta'_2 &= \theta_2 + N(\mu_{rot}, \sigma_{rot}, RANDOM) \\ d' &= d + N(\mu_{trans}, \sigma_{trans}, RANDOM) \end{aligned}$$

| Sample Number | Average Error in distance conversion |
|---|---|
| 1 | 0.602492 |
| 2 | 0.904399 |
| 3 | 1.08077 |
| 4 | 1.14689 |
| 5 | 0.873715 |
| 6 | 1.07968 |

TABLE I: Average error in distance conversion over various trials

where RANDOM denotes a random sample drawn from the Gaussian, and the New state - $X' = [x' \; y' \; \theta']^T$ is given by

$$\begin{aligned} x' &= x + d' * \cos(\theta + \theta_1'') \\ y' &= y + d' * \sin(\theta + \theta_1') \\ \theta' &= \theta + \theta_1' + \theta_2' \end{aligned}$$

### E. Measurement Model

This step weights a given particle $i$ according to the likelihood function $p(z_k|x^i)$. The measurement vector is the trilaterated output $[x \; y]^T$. The likelihood function is modeled as a Gaussian centered around this point whose variance is estimated empirically, using static localization. For the Measurement vector - $z = [x \; y]^T$ and a particle $X = \{x^i, y^i, w^i\}$, the update equation [14] is given by,

$$w'^i = w^i N(x, \sigma_x, x^i) N(y, \sigma_y, y^i)$$

Since the bluetooth dongles used do not provide any angle information like angle of arrival, $\theta$ does not appear in the update equation.

## V. Experimental Results

A 6m X 8m floor was used, part of which was cluttered with furniture. Three bluetooth beacons(USB dongles) were fixed at three known coordinates. The robot was a two-wheeled differential drive robot. No other sensor information or odometry was available. The robot performed a series of random translations followed by one random rotation. The rotations were multiples of $\frac{\pi}{2}$ radians. It performed 5 inquiries after each motion using the dongle mounted on it and the average RSSI was taken. The inquiries were short inquiries with timeout of 3s. The errors are calculated based on ground truth measurements.

### A. Static localization

RSSI values observed by the robot are mapped onto a set of possible distances as discussed in section 2.B. The average error in the distance mapping for various runs is shown in Table I. Clearly, the error in RSSI-distance conversion is quite large.This error is due to inherent noise in RSSI measurement. The mapping is not one-to-one, and all the possible distances are taken into account.

All possible distances from the above step are taken as input for trilateration. The iterative trilateration error proved to be a tremendous improvement over pseudo-inverse trilateration. Shown below in Fig. 4 is a scenario from one of the runs. The beacon positions are $(1.72, 0.35), (1.79, 5.63), (3.05, 2.61)$ and the corresponding measured distances are $0.752835, 4.4$ and $3.07207$. The robots actual position was $(1.05, 0.79$. The normal trilateration solution was $(-1.2049, 3.7668)$ , whereas the iterative method resulted in $(1.0306, 0.7756)$ which is much more accurate.

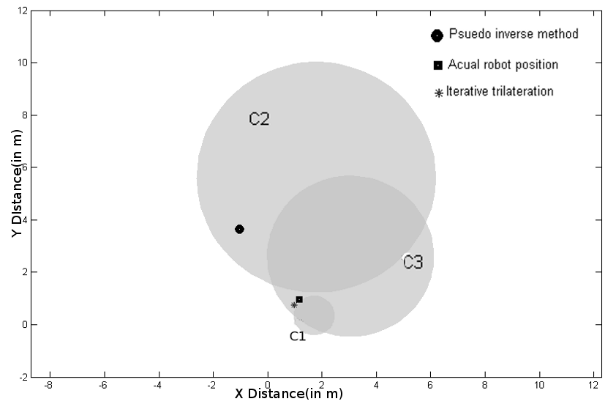


Fig. 4: Comparison of iterative trilateration and pseudo-inverse trilateration

The error in mapping as well as trilateration contribute to the static localization error. This error is compared with the error after particle filtering in Fig. 5.

### B. Particle Filter

Fig. 5 shows the error over time of one of the trials. The error after using the particle filter is compared with the trilateration error in each particular iteration. This graph shows the robustness of the system towards outliers i.e. when the trilateration error is large.

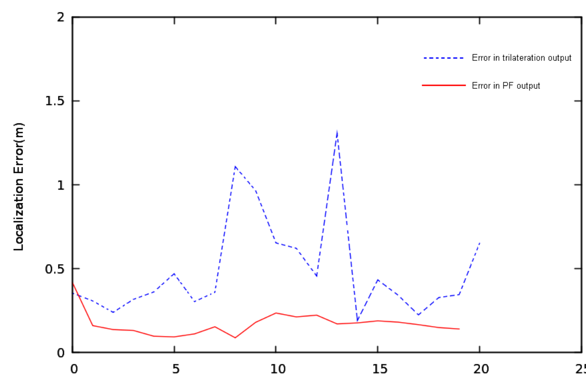


Fig. 5: Comparison of error with and without the particle filter.

### C. Overall Results

The results of various trials are shown in the table II. The mean error was observed to be **0.427m** with a standard deviation of 0.229m. It can be seen that the algorithm performs better in the proximity of more number of beacons, but still gives acceptable errors of less than $< 1m$ in highly

| Run No. | $NI$ | $k$ | $\varepsilon_f$ | Comments |
|---|---|---|---|---|
| 1 | 9 | 15.45 | 0.20 | No furniture |
| 2 | 17 | 59.06 | 0.78 | Amidst furniture |
| 3 | 12 | 13.83 | 0.30 | Centre of 2 beacons |
| 4 | 23 | 11.22 | 0.63 | Near beacon |
| 5 | 18 | 12.60 | 0.15 | Centre of 3 beacons |
| 6 | 14 | 21.35 | 0.50 | Farthest from beacons |

TABLE II: This table shows the results of the experiment: $NI$ - No. of iterations, $k$ - Avg. No. of trilaterations, $\varepsilon_f$ - final error(m), Comments - about the starting location. The number of iterations are different for each trial simply because the robot had reached the boundary of the arena due its purely random motion.

cluttered areas. In most other RF based methods positioning errors increase dramatically in cluttered indoor environments, but our algorithm still performs reasonably. The locations of the beacons were known before each trial. It was observed that when the robot starts from or moves into an *unfavorable* location, the number of distance mappings and the average number of trilaterations increase. An *unfavorable* position is one in which the the measured RSSI is an outlier. Motion planning techniques such as, moving towards the centroid of the beacons(say), can be explored. It would reduce the error as well as the number of iterations. This becomes crucial when the environment has a lot of obstacles.

## VI. Conclusions and Future Work

In this paper we show that bluetooth can be used for robot localization in indoor environments with an accuracy of $< 1m$ using a computationally inexpensive method. In spite of the Gaussian assumptions and lack of odometry, we achieved an accuracy of **0.427 ± 0.229m**. Except the locations of the beacons, an extensive knowledge of the environment is not required. Limitations as cited by previous works have been overcome. The time for each iteration is dominated by the time required to perform inquiry. This can be improved by the use of *Interlaced inquiry*. Presence of obstacles in the environment affects the performance of our system. It can be seen that some locations are *favorable* and planning the motion accordingly would result in better results. We also plan to explore the use of auxiliary particle filters and class 1 dongles to obtain better performance. Although we chose Bluetooth, this system can be implemented any wireless technology that provides an RSSI value.

## VII. Acknowledgments

We acknowledge Balaji Lakshmanan from the RISE lab,Indian Institute of Technology,Madras for developing the 'MoBo' mobile robot platform that was used in the experiments.

## References

[1] J. Hallberg, M. Nilsson, and K. Synnes. Positioning with bluetooth. In ICT 2003: *10th International Conference on Telecommunications*, volume 2, pages 954-958 vol.2, Feb- March 2003.

[2] A. Madhavapeddy and A. Tse, *A Study of Bluetooth Propagation Using Accurate Indoor Location Mapping*, 2005, vol. 3660. [Online]. Available: http://www.metapress.com/content/2FGWHFACYGC011HY

[3] Timothy M. Bielawa, Position Location of Remote Bluetooth Devices, *MS Thesis, Virginia Polytechnic Institute and State University, 2005*

[4] Kotanen A., Hannikainen M., Leppakoski H. and Hamalainen T.D, Experiments on local positioning with Bluetooth. In ITCC 2003: *International Conference on Information Technology: Coding and Computing [Computers and Communications], 2003. Proceedings*, pages 297-303, 28-30 April 2003.

[5] Silke Feldmann, Kyandoghere Kyamakya, Ana Zapater, and Zighuo Lue. An indoor bluetooth-based positioning system: Concept, implementation and experimental evaluation. In International Conference on Wireless Networks, pages 109-113, 2003.

[6] Varun Almaula and David Cheng, Bluetooth Triangulator, *Final Project, Department of Computer Science and Engineering, University of California, San Diego, 2006*

[7] Sheng Zhou and J.K. Pollard. Position measurement using bluetooth, *IEEE Transactions on Consumer Electronics*, 52(2):555-558, May 2006.

[8] Specification of the Bluetooth System, Version 2.0 + EDR, Bluetooth SIG, 2004

[9] Albert Huang and Larry Rudolph, A Privacy Conscious Bluetooth Infrastructure for Location Aware Computing, *Massachusetts Institute of Technology* 2004.

[10] Bing-Fei Wu, Cheng-Lung Jen, and Kuei-Chung Chang. Neural fuzzy based indoor localization by Kalman filtering with propagation channel modeling. *In IEEE International Conference on Systems, Man and Cybernetics, 2007* , pages 812-817, Oct. 2007.

[11] Erin-Ee-Lin Lau and Wan-Young Chung. Enhanced RSSI-based realtime user location tracking system for indoor and outdoor environments.*In ICCIT 07: Proceedings of the 2007 International Conference on Convergence Information Technology* , pages 1213-1218, Washington, DC, USA, 2007. IEEE Computer Society.

[12] Sebastian Thrun, Wolfram Burgard and Dieter Fox, Probabilistic Robotics, September 2005

[13] Zhe Chen, Bayesian filtering: From Kalman filters to particle filters, and beyond, *Technical report, McMaster University, 2003*.

[14] Ioannis M. Rekleitis, A particle filter tutorial for mobile robot localization, *Technical Report TR-CIM-04-02, Centre for Intelligent Machines, McGill University, 3480 University St., Montreal, Quebec, Canada H3A 2A7, 2004*.

[15] J. Carpenter, P. Clifford, and P. Fernhead, An improved particle filter for non-linear problems,*tech. rep., Department of Statistics, University of Oxford, 1997*.