International Journal of Computer Applications Technology and Research
Volume 2– Issue 6, 690 - 698, 2013, ISSN: 2319–8656

The proposed System for Indoor Location Tracking

Ashraf Mohamed Ali Hassen
Department of Electronics and Communications
Modern Academy in Maadi, Egypt

Abstract: Indoor location tracking systems are used to locate people or certain objects in buildings and in closed areas. For example, finding co-workers in a large office building, locating customers within a shopping mall and locating patients in the hospital are a few applications of indoor location tracking systems. Indoor tracking capability opens up multiple possibilities. To address this need, this paper describes the implementation of a Bluetooth-based indoor location tracking system that utilizes the integrated Bluetooth modules in any today's mobile phones to specify and display the location of the individuals in a certain building. The proposed system aims for location tracking/monitoring and marketing applications for whom want to locate individuals carrying mobile phones and advertise products and services.
Keywords: Bluetooth, RSSI, Indoor tracking, location aware applications, ISM Band, ubiquitous, GPS

1. INTRODUCTION

Today's, Bluetooth functionality is widely included in almost all mobile phones. As a result most people carrying a mobile phone are also carrying a Bluetooth capable device. Hence, using the Bluetooth protocol contributes in minimizing the deployment cost of an indoor localization system. A Bluetooth-based indoor tracking system using low-cost stationary beacons and mobile host devices is described. The system permits low-cost, rapid deploying of applications where room-level location detection is sufficient. To allow the proposed localization algorithm to be applied to a variety of mobile phones, the algorithm is executed on a central server instead of the mobile phone and the location information resides on the central server. The proposed system is composed of a Server Module which is a java application that runs over desktop PC and is used to display the locations of the nearby mobile phones and send location-based advertising/control message/signal. The system also composed of a Locating Modules which written in C and runs over the Bluetooth-enabled boards which used to periodically collect the Bluetooth addresses of the nearby devices/mobiles accompanied with their sensed signal strengths and send them to the Server Module to process, specify, and display the location of the mobile phones. In this experimental study there are four Bluetooth-enabled boards used as prove of concept.

2. RELATED WORK

Indoor location tracking systems deliver usefulness to end users, if designed and maintained appropriately. The interest for assistive technology is increasing. Generally the need for services increases with age [1]. It is important to note that there is no one perfect location system. Each system must be evaluated based on the intended application across a variety of dimensions such as its accuracy, the infrastructure requirements, and the ability to scale.

2.1 Active Badge

The Active Badge is designed to be worn by visitors and employees of an organization to allow a central database to keep track of their location within the building. The badge transmits a unique code via a pulse-width modulated IR signal to networked sensors/receivers deployed throughout a building. The Active Badge uses 48-bit ID codes and is capable of two-way communication. The badge periodically beacons the unique code (approximately every 10–15s), and the information regarding which sensors detected this signal is stored in a central database. Since the IR signal does not travel through wall, the sensors are deployed throughout the space [2]. As with any diffuse infrared system, Active Badges have difficulty in locations with fluorescent lighting or direct sunlight because of the spurious infrared emissions these light sources generate.

2.2 Active Bat

Active Bat [3] is an ultrasonic-based location tracking systems consisting of ultrasound receivers dispersed in a space and location tags that emit ultrasonic pulses. Active Bat tags emit short pulses of ultrasound and are detected by receivers mounted at known points on the ceiling, which measure the time-of-flight of each pulse. Using the speed of sound, the distance from the tag to each receiver is calculated. Given three or more measurements to the receivers, the position of the tag can be determined using trilateration. One key concept of Active Bat is the use of an RF signal to cue the tag to transmit its ultrasonic pulse. The RF cue gives the receivers in the environment a starting point for timing the received ultrasonic pulse. Since the speed of light is significantly faster than the speed of sound, the RF signal delay is negligible and does not need to be considered for calculating the time-of-flight of the acoustical signal. The information about the location of Active Bat tags is managed by a central server. Using ultrasound time of flight this way requires a large fixed-sensor infrastructure throughout the ceiling and is rather sensitive to the precise placement of these sensors. Thus, scalability, ease of deployment, and cost are disadvantages of this approach [2].

3. SYSTEM DESCRIPTION

The proposed system utilizes the integrated Bluetooth modules in any today's mobile phones to specify and display the location of the individuals in a certain building. The proposed system aim for smart home location tracking/monitoring applications and marketing applications for whom want to locate individuals carrying mobile phones and provide context services and/or products advertisements. It is an integrated embedded and desktop system that helps the user to get the location of customers/home inhabitants within a certain region where there are several Locating Modules that are installed as one node per room to cover certain area which periodically investigate/inquiry the Bluetooth signal strength and addresses of the nearby mobile phones and send these data to a central Server Module. The Server Module

processes, specifies, and displays the location of each mobile phone, by means of its MAC address using table and a simplified graphical map. Moreover the administrator can set certain messages to be sent if the mobile phone is near a certain place, where the Server Module can send the settled message to the mobile users.

The proposed approach is motivated by the drawbacks reported by the existing indoor localization approach and by the resource limitations typically imposed on the simple Bluetooth nodes, where a neural network-based scheme is described. The proposed algorithms based mainly on measurement of the received signal strength indicator (RSSI) this measurement is used as input to an algorithm that estimates the location of the mobile node that algorithm utilizes the use of neural network rather than specific measurement values and it targets the same order of simplicity as the I^{3}BM [4] algorithm, but with the high accuracy of the Environment Adaptive [5] [6] algorithm. To do so; it includes a training phase, for modeling the surrounding environment and calculating the internal weights of the neural network, and location determination phase. Room-level location is selected, where rooms refer to physical rooms or specific logical subspaces, as this is the most needed for smart home applications or generally indoor environment.

4. SYSTEM ARCHITECTURE

The system architecture is shown in figure 1

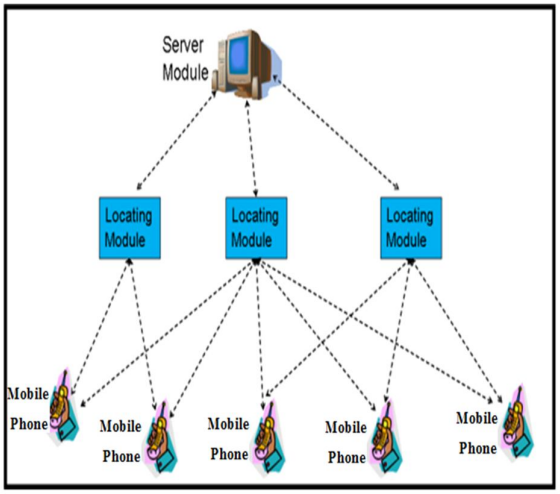

Figure 1 system architecture

4.1 Server Module: It is a JAVA application runs over desktop personal computer and it is used to process, specify and display the locations of the nearby mobile phones and send location-based advertising message. This application utilizes a Java neural network library, by Neuroph, that supports creating, training and saving neural networks.

4.2 Locating Modules: It is a C program that runs over the Bluetooth-enabled Arduino boards, used to periodically collect the Bluetooth addresses of the nearby devices accompanied with their signal strengths and send them to the Server Module to process, specify, and display the location of the mobile phones. Each Locating Module covers a number of nearby mobile phones according to their location and received signal strength. In this study there are four Bluetooth-enabled boards used as a prove of concept. Figure 2 shows rooms layout.

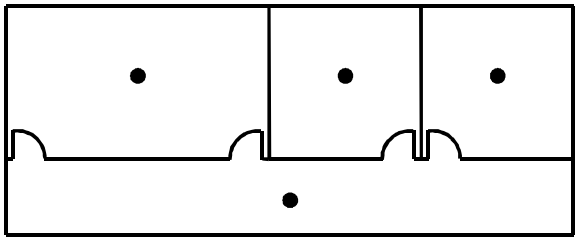

Figure 2 rooms layout

4.3 Mobile phones with Bluetooth service enabled and set to visible. The administrator will just invoke the application and communicate with it appropriately to explore nearby mobile phones.

5. TRACKING ALGORITHM

Indoor location tracking/monitoring approach based mainly on measurement of signal characteristic as the received signal strength indicator (RSSI), the link quality indicator (LQI), the time of arrival (ToA), or the angle of arrival (AoA). This measurement is used as input to an algorithm that estimates the location of the mobile node; in this study we select RSSI as our measure. Measuring RSSI is very simple, it is implemented in all Bluetooth protocol stacks and is available for all Bluetooth platforms. Hence, there are many indoor location monitoring/tracking schemes that utilize this metric as the measurement for location estimation [7]. Among the most popular are the I^{3}BM [4] and the Environment Adaptive [1] [5].

The major challenge, which has addressed by all indoor location monitoring/tracking schemes, is the high sensitivity of the measured signal characteristic to the surrounding environment. The I3BM scheme has addressed this challenge through mechanisms as signal filtering and room adjacency utilization. The scheme has maintained simplicity, but the achieved accuracy has not been that acceptable [4]. The Environment Adaptive has addressed the high sensitivity challenge by modeling the environment in a calibration phase. Accurate location has been obtained at the expense of high computational cost and large memory requirement [5] [6].

5.1 I^{3}BM Algorithm

The I^{3}BM targets room-level position estimation. It positions the mobile node in the room whose beacon node yields the highest RSSI. To reduce the high sensitivity of the measured values to the surrounding environment, the I3BM employs mechanisms as signal filtering and room adjacency utilization. As there are many factors influencing the signal readings, it is irrational to base the positioning decision upon a single measurement, but rather upon the outcome of filtering a group of signal values [4] [8].

Assuming that noises tend mainly to reduce the signal strength, a MAX filter has demonstrated good performance, where the selected RSSI value is the maximum received value for a sequence of packets communicated between the mobile and each beacon node over a pre-specified period of time[8]. Room adjacency utilization also improves accuracy of location estimation where people move according to feasible paths. Hence, only rooms that are adjacent to current location should be considered as possible estimate for the new location. Figure 8 demonstrates the room adjacency relationship for the layout in Figure 2.

The I^{3}BM scheme is simple but it doesn't provide enough accuracy to build a convenient smart home application. This is due to the fact that the signal measurements are highly

sensitive to the surrounding environment and only the information contained in the highest RSSI is utilized for decision making, whereas all the information reported by the other beacon nodes are ignored.

5.2 Environment Adaptive Algorithm

The Environment Adaptive scheme has been developed [5] [6] driven by the goal of having an accurate position estimation scheme that is robust to environmental changes. The scheme has two phases, calibration phase and localization phase. In the calibration phase, environment characteristics are modeled by computing the m × m transformation matrix, T, which relates the distance between the m anchor nodes and the RSSI of the packets communicated between them using

$$T = D\,S^{-1} \quad \text{--- (1)}$$

Where D denotes the m × m distance matrix whose element d_{ij} represents the known Euclidean distance between the i^{th} and j^{th} anchor nodes, and S denotes the m × m strength matrix whose element s_{ij} represents the RSSI value of the packets between the i^{th} and j^{th} anchor nodes.

In the localization phase, given T and the m-dimensional vector I, whose element I_i represents the RSSI value of the packets between the mobile node and the i^{th} anchor node, then

$$V = T\,I \quad \text{--- (2)}$$

Where V denotes the m-dimensional vector whose element v_i is the calculated distance between the mobile node and the i^{th} anchor node. The mobile node position estimate, P(x, y) is then determined using the following gradient descent equation [9] [10].

$$P_{k+1} = P_k + \propto \sum_{i=1}^{m} \left(1 - \frac{V_i}{E(A_i, P_k)}\right)(A_i - P_k) \quad \text{--- (3)}$$

where P_k is the mobile node position estimate at the k^{th} iteration, A_i is the position of the i^{th} anchor node, α is the gradient step size, and $E(A_i, P_k)$ is the Euclidean distance between the i^{th} anchor node and the mobile node position estimate. At the first iteration, the scheme assumes that the mobile node is directly located at the position of the anchor node with the minimum v_i. The estimated mobile node position, (x, y), is finally mapped to a specific room as per the space dimensions of the underlying environment.

The Environment Adaptive scheme has demonstrated good performance and much less sensitivity to variations in the surrounding environment [5]. However, the inverse in (1) and the iterative nature in (3) are computationally expensive and necessitate slow response and large memory requirements. Here, it should be noticed that the scheme can utilize either the RSSI measurements or its LQI alternative; the original description of the scheme [5] employs the RSSI measure.

5.3 The Proposed Neural Network-based Scheme

Neural networks are mostly used for fuzzy, difficult problems that don't yield to traditional algorithmic approaches. Learning in neural networks is particularly useful in applications where the complexity of the data or task makes the design of such functions by hand impractical. The proposed system utilize a neural network which could be described as "4 × 6 × 4 neural network" where it has one input layer with four neurons, one hidden layer with six neurons and one output layer with four output neurons.

5.3.1 Developing Environmental Radio Map

The first step is creating a radio map of the area. Measurement locations were chosen roughly every one meter, as shown in figure 3, excluding the places that were covered by tables or desks. The measurements were taken by a person who was holding the mobile phone, to simulate real usage. Because Bluetooth signals are absorbed by the human body, the direction of the person with regard to the beacon can influence the signal. We therefore measured the signals in four different directions, taking 1 sample in each direction (facing north, east, west and south). Overall measurements were taken on 152 different locations, giving a total of 608 fingerprints.

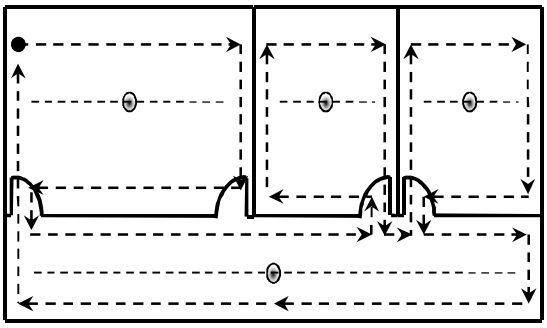

Figure 3 Radio Map for the locations of measurements

These measurements are applied to a neural network within the JAVA program as a training set. Primarily we utilized a tool named "Neuroph" which is a lightweight Java neural network framework to develop common neural network architectures. It contains well designed, open source Java library with small number of basic classes which correspond to basic neural network concepts. Also has nice GUI neural network editor to quickly create Java neural network components. Creating the neural network architecture therefore means coming up with values for the number of layers of each type and the number of nodes in each of these layers.

Every neural network has exactly one input layer; and the number of neurons comprising this layer is completely and uniquely determined as the number of columns in your data which is four in our case. Also additional node as a bias node is included, Bias neuron is very important, and the error-back propagation neural network without Bias neuron does not learn. A bias input always has the value of 1. Without a bias, if all inputs are 0, the only output ever possible will be a zero [11].

Like the Input layer, every neural network has exactly one output layer. Determining its size (number of neurons) is simple; it is completely determined by the chosen model configuration. In this experiment the neural network is used as a classifier, and then it has number of output neurons equal to four which is the number of rooms to classify between them [11].

Finally, concerning the hidden layers since one hidden layer is sufficient for the large majority of problems where there is a consensus is the performance difference from adding additional hidden layers: the situations in which performance improves with a second (or third, etc.) hidden layer are very small. Now to determine the correct number of neurons to use

in the hidden layers several trails is performed that guided by the following rule-of-thumb [11]:

- The number of hidden neurons should be between the size of the input layer and the size of the output layer.
- The number of hidden neurons should be 2/3 the size of the input layer, plus the size of the output layer.
- The number of hidden neurons should be less than twice the size of the input layer.

These three rules provide a starting point for our guided trial and error process, with the help of the visual tool "Neuroph Studio"; we reach to the optimal size of hidden layer to be six neurons. Finally we can describe the utilized neural network as "4 × 6 × 4 neural network" where it has one input layer with four neurons, one hidden layer with six neurons and one output layer with four output neurons.

5.3.2 Procedure of Obtaining a Representative Trained Neural Network

In order to train neural network these data set have to be normalized. Normalization implies that all values from the data set should take values in the range from 0 to 1. For that purpose it would be used the following formula:

$$X_n = \frac{X - X_{max}}{X_{max} - X_{min}}$$

Where:

X, is the value that should be normalized

X_n, is the normalized value

X_{min}, is the minimum value of X

X_{max}, is the maximum value of X

The output is represented by 4 digits of data set represent the target room. For example, "1 0 0 0" represent office room, "0 1 0 0" meeting room, "0 0 1 0" lab room and "0 0 0 1" cafeteria room.

We used the Neuroph [11] wizard to create a Neuroph project, during this wizard there are some settings that we used to create the neural network. Among these setting is the type of training, which is set to supervised training which is the most common way of neural network training. As supervised training proceeds, the neural network is taken through a number of iterations, until the output of the neural network matches the anticipated output, with a reasonably small rate of the error. Figure 4 shows Trained Neural Network Snapshoot.

To achieve the main goal and decide which the best solution for this problem is, several neural networks are created with different settings. Among these settings is the number of neurons in the hidden layer which play a vital role in the whole system performance. Five, six, seven and eight neurons in the hidden layers are tried and the best performance is estimated and presented in the results section.

For testing the neural network, the experimental data were divided randomly into two sets (training and testing dataset). Separation of 70/30 dataset was done for training/testing where 70% of the experimental data used to train the neural network and 30% of it for testing. Simply Neuroph framework is used to test the performance of the trained network.

This is repeated several times within Neuroph framework till reaching to the optimal results then the corresponding parameters are utilized in the final Java application; experimental results are given in what follows.

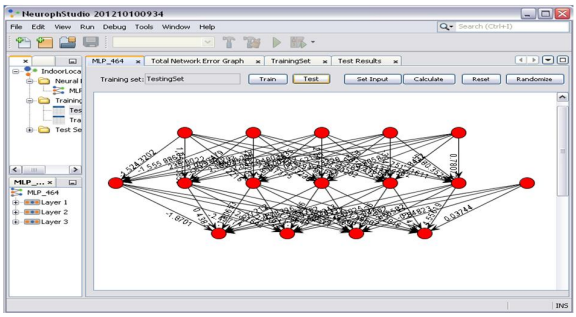

Figure 4 Trained Neural Network Snapshoot

6. SYSTEM CONFIGURATION

6.1 Locating Modules Setup

Setup one locating module in each room, it is recommended to have the locating modules separated by minimum of 4 meters and maximum of 10 meters.

6.1.1 Arduino BT (Bluetooth) Board

The Arduino BT shown in figure 5 is a microcontroller board based on the ATmega328 and the Bluegiga WT11 Bluetooth module[12].

Figure 5 Arduino Board

It supports wireless serial communication over Bluetooth (but is not compatible with Bluetooth headsets or other audio devices). It has 14 digital input/output pins (of which 6 can be used as PWM outputs and one can be used to reset the WT11 module), 6 analog inputs, a 16 MHz crystal oscillator, screw terminals for power, an ICSP header, and a reset button. It contains everything needed to support the microcontroller and can be programmed wirelessly over the Bluetooth connection. Each of the 14 digital pins on the Arduino BT can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions.

6.2 Server Module Setup

A Computer device that has a Bluetooth dongle connected to it through the USB port. The computer device run the developed JAVA application, named Server Module, which

could connect to all Locating Modules to sent command and receive response. Figure 6 shows example of server measurements configuration file.

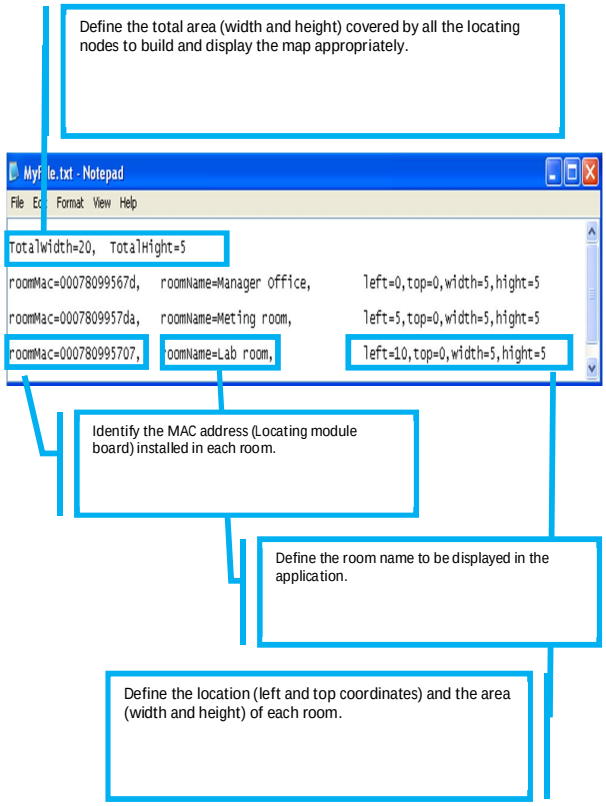

Figure 6 Example of server measurements configuration file

The running java application has a suitable UI interface, which allow for users to easily benefits from different capabilities of the system.

6.2.1 Building the Server Configuration File

The configuration file will be located on the administrator module machine. It is just a text file contains in its first line the total width and total height of the area that the map will displayed in, then each line next the first line contains a complete entry. Any entry will contains room MAC, room displayed name, room top position, room left position, room width and finally room height. The server configuration file is needed to inform the server of the mapping between the locating module and the corresponding room, the following example demonstrates the contents of the file.

6.3 System Hardware Requirements

Table 1. Hardware Resources

Resources Name	Description
Mobile Phone	Any mobile phone that supports Bluetooth and Messaging
Arduino Board	Bluetooth Enabled board
ATMEGA328P Programmer	To burn the Arduino application on the board
Server PC	To deploy/run Server Module
USB Bluetooth Dongle	Attached to the Server PC

6.4 System Software Requirements

Table 2. Software Resources

Resources Name and version	Description
NetBeans IDE	Used for developing the Server Module application
Arduino IDE	Used to develop and compile the application that run on Arduino Bluetooth Board
AVR Studio 4	Used to program the Atmeg328P microcontroller that is on Arduino Bluetooth Board

7. SYSTEM DEPLOYMENT

Figure 7 depicts the layout of the environment used to develop the indoor locating system. As depicted, a single fixed, beacon node is installed in each room at a pre-determined location.

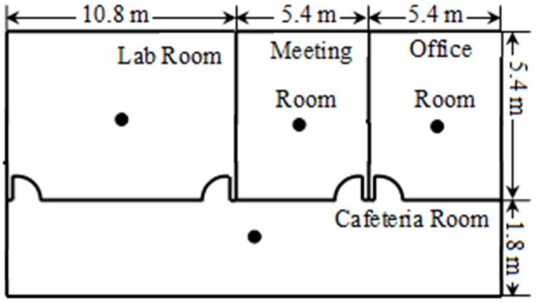

Figure 7 Layout of the overall system

The experiments were carried out using Arduino Bluetooth board[12] as beacons where there is a board installed per each room, and mobile phones from different brands (Nokia 5630 Music Express, BlackBerry 9700, and Samsung D900i) for measuring the Bluetooth signals. There is no software needed to pre-install on the mobile phones just it is required to be Bluetooth enabled and made discoverable. The physical environment consist of four rooms where there is typically three physical rooms titled as Office room, Meeting room, Lab room and finally the corridor along these rooms but virtually we considered it as another room named, Cafeteria room, as shown in figure 7. The office room has size of 5.4x5.4 m2 large, the meeting room has size of 5.4x5.4 m2, the Lab room has size of 10x5.4 m2 and the corridor has size of 21.6x1.8 m2 large. The Locating Module run over the Arduino Bluetooth-enabled boards/beacons and the Server Module runs over a personal computer with a Bluetooth dongle. It should be noticed that beacon nodes are placed such that once the mobile node is moved into a room it becomes closest to that room's beacon node, taking into account the effect of walls separating different rooms. This is rational to probabilistically increase the RSSI value of the signal received from the closest beacon node over the values received from the others.

Server module will create one thread to collect the data from all locating modules (4 modules in this study). By this approach the server module will avoid synchronization problems on shared resources between multiple threads. But will add more code complexity to minimize the refresh rate. Within this thread the server Module will open multiple Bluetooth connections simultaneously with multiple locating

modules using the Bluetooth dongle and periodically iterate on all the boards to send command and receive response.

The server module will iterate periodically on all the boards, each iteration composed of two rounds, in the first round the server module will just send start "S" command to all the boards sequentially. In the second round, the server module might wait only once at the first board to finalize its inquiry time but all other following boards will be ready with its inquiry response and the server module will retrieve its data without waiting more time.

When the locating module receives the start, "S", command it will start the inquiry for nearby mobiles. Upon receiving this inquiry by nearby mobiles, each mobile node will reply that inquiry by an inquiry response that contains the MAC address of that mobile and its corresponding signal strength, RSSI. The RSSI value in the inquiry response is automatically calculated by the Bluetooth MAC layer. The locating module will collect these data (MAC addresses and signal strength) and send it line by line to the server module just upon server module read operation. When the locating modules send all the data to the server module the locating module will be waiting for the next start command. The locating module will close the connection with the server module upon user request to exit/stop the application where the server module will send a stop command "X" to all beacon nodes that upon receiving the stop command will close the connection with the server module.

Now the server module processes the whole information and estimates the locations of all Bluetooth-enabled mobile phones. It then invokes the appropriate services for each mobile user according to the estimated location. Actually the Server module employ a trained neural network where for a certain mobile MAC address there will be a vector R that has the RSSI values of that MAC address for each room where r_i refer to the RSSI value of that mobile as reported by the beacon node in room i. by applying this as an input to the trained neural network and based on the different values of the internal weights between neurons that reflect the environmental effect on the measured value some output neurons are activated referring to the estimated location (room-level). Figure 8 shows room adjacency relationships

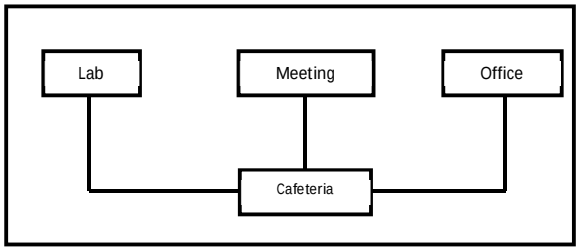

Figure 8 ROOM Adjacency Relationships

To improve the accuracy of position estimation a post processing mechanism as room adjacency is utilized where people move according to feasible paths. Hence, only rooms that are adjacent to current position should be considered as possible estimate for the new position.

To improve the results of the overall system and enhance system stability of the displayed status, the mobile node has to say two consecutive times that it is say in room 1 but this will affect the refresh rate of the displayed status so a confidence factor is calculated and this value is compared to certain confidence threshold, calculated by trial and error, and based on this comparison the displayed status will be updated minimizing the refresh rate and improving the system stability.

8. SYSTEM TEST RESULTS

Figures 9 ,10,11 and 12 show total network error using eight,five,six and seven neurons respectively tables 3,4,5 and 6 Show System Positioning Estimation Accuracy Using Eight ,five ,six and seven Neurons respectively these results are taken through different attempts.

8.1 Attempt 1

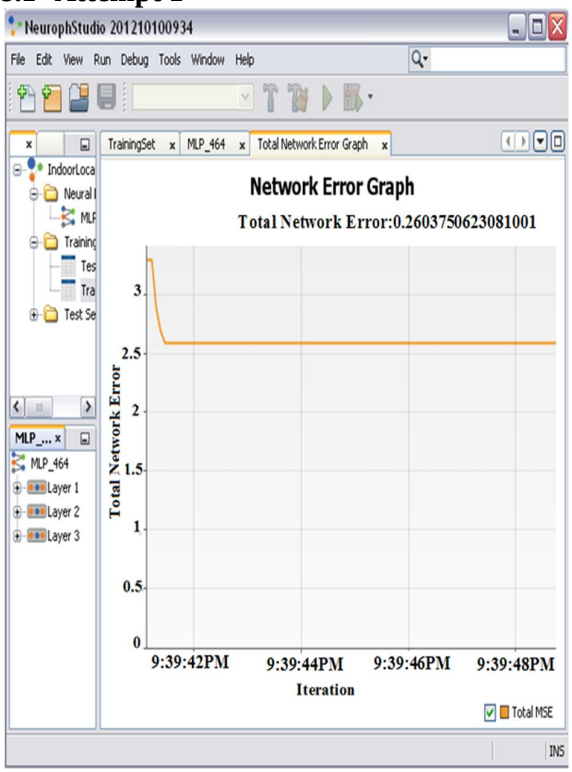

Figure 9 Total Network Error Using Eight Neurons

Table 3. System Positioning Estimation Accuracy Using Eight Neurons

Scheme	Refresh Rate	Accuracy
Neural network utilized	10 sec	73.93%
Room adjacency applied	10 sec	87.23%
Room adjacency applied and twice per room	20 sec	91.48%
Room adjacency and twice per room with confidence threshold applied	10 sec	91.48%

Using Neuroph, We create a neural network with 4 input neurons, 4 output neurons and set the number of neurons in the hidden layer to eight. Using the collected training data samples to train the neural network, and after testing we got that in this attempt the total network error is 26% as illustrated in the figure. Now utilizing this neural network in our java application with these configurations we get the following results.

8.2 Attempt 2

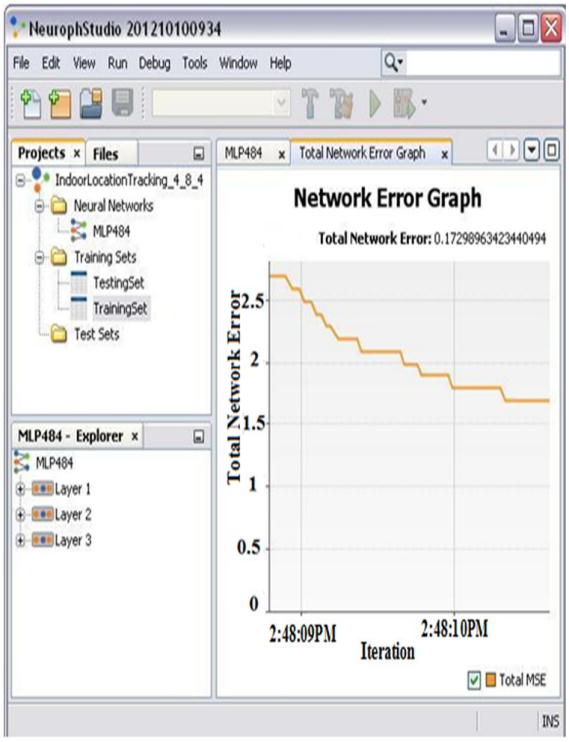

Figure 10 Total Network Error Using Five Neurons

Table 4. System Positioning Estimation Accuracy Using Five Neurons

Scheme	Refresh Rate	Accuracy
Neural network utilized	10 sec	82.97%
Room adjacency applied	10 sec	90.95%
Room adjacency applied and twice per room	20 sec	94.68%
Room adjacency and twice per room with confidence threshold applied	10 sec	94.68%

We create a new neural network with 4 input neurons, 4 output neurons and set the number of neurons in the hidden layer to five. Using the same collected training samples as previous attempt to train the neural network, after testing we got that in this attempt the total network error is 17% as illustrated in the figure. Now utilizing this neural network in our java application with these configurations we get the following results.

8.3 Attempt 3

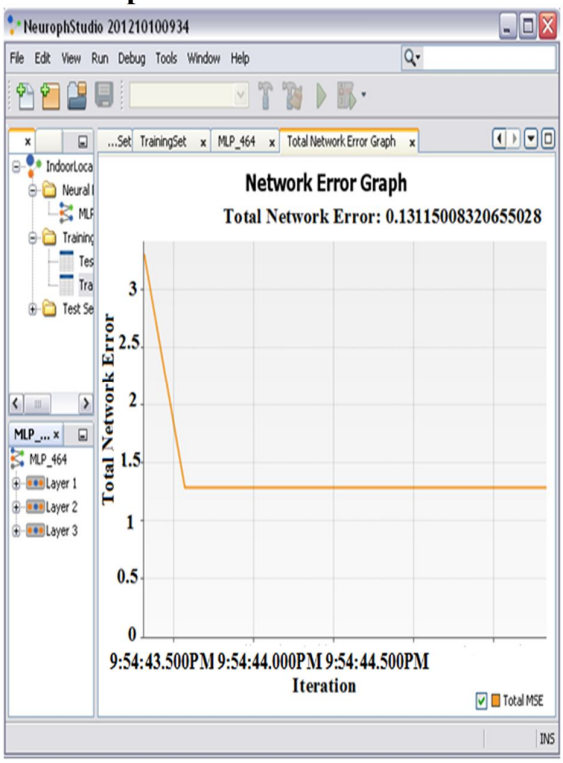

Figure 11 Total Network Error Using Six Neurons

Table 5. System positioning estimation accuracy using six neurons

Scheme	Refresh Rate	Accuracy
Neural network utilized	10 sec	86.70%
Room adjacency applied	10 sec	95.21 %
Room adjacency applied and twice per room	20 sec	97.34%
Room adjacency and twice per room with confidence threshold applied	10 sec	97.34%

We create a new neural network with 4 input neurons, 4 output neurons and set the number of neurons in the hidden layer to six. Using the collected training samples, as in the previous attempts, to train the neural network, after testing we got that in this attempt the total network error is 13% as illustrated in the figure. Now utilizing this neural network in our java application with these configurations we get the following results.

8.4 Attempt 4

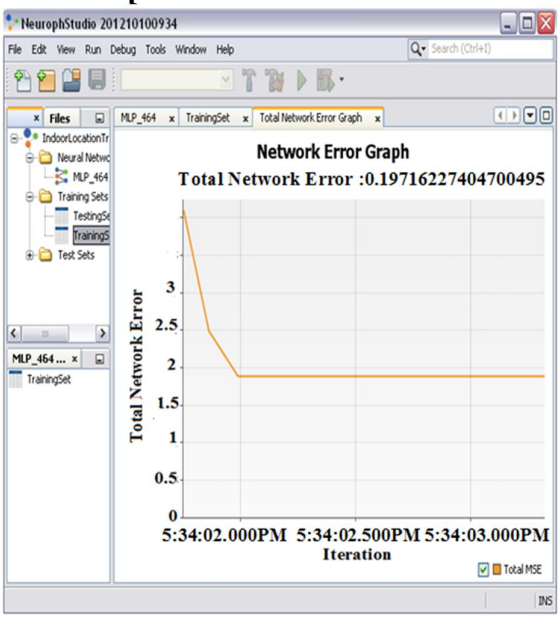

Figure 12 Total Network Error Using seven Neurons

Table 6. System positioning estimation accuracy using seven neurons

Scheme	Refresh Rate	Accuracy
Neural network utilized	10 sec	80.32%
Room adjacency applied	10 sec	89.36%
Room adjacency applied and twice per room	20 sec	92.02%
Room adjacency and twice per room with confidence threshold applied	10 sec	92.02%

Finally, we create a new neural network with 4 input neurons, 4 output neurons and set the number of neurons in the hidden layer to seven. Using the collected training samples, as in the previous attempts, to train the neural network, after testing we got that in this attempt the total network error is 19% as illustrated in the figure. Now utilizing this neural network in our java application with these configurations we get the following results.

9. CONCLUSION

As demonstrated, proper selection of the number of internal neurons in the hidden layer influences the accuracy of the results and from the mentioned trials we could conclude that the optimum size of the hidden layer in this typical environment is six neurons that gave the optimum results. Another point utilizing room adjacency relationships consistently improved estimation accuracy. Also, to minimize the stochastic instability of the RSSI measurements the user location is updated only when the system report two consecutive times the same decision but this effect the system refresh rate and to retain the refresh rate again to its original value a confidence threshold value is applied and by this approach estimation accuracy and system stability are maximized. In a nutshell estimating location is based not only upon a single value reading of the RSSI from different rooms and selecting the highest RSSI, but based on the nature of the underlying environment which implicitly included in the neural network weights calculated in the training phase and applying a post processing fixes to enhance system accuracy and stability.

10. FUTURE WORK

Still, the field of indoor location tracking is not yet mature enough to propose a universal solution that works in any setting. Currently, the choice of technology and positioning algorithm depends mainly on the requirements of the system. One important aspect to consider is how our buildings will be used in the future. Will we perform other or different daily activities other than today? Do we need support from technology in order to perform such new activities?

Regarding the technology, multiple technologies have been proposed to tackle the problem of indoor localization some examples being infrared, ultrasound and Radio Frequency IDentification. Still, most research is dedicated to the usability of two technologies. Large number of papers, e.g., [13] and [14], argue that Ultra Wide Band (UWB) radio offers excellent means to determine one's location with high precision. To summarize, there is no single location technology today that is ubiquitous, accurate, low-cost (in terms of required hardware and installation), and easy to deploy. Although a novel location technology that fits all these parameters might still become available in the future, a more realistic approach is to combine several of the existing technologies into an integrated location system. For instance, integration between GPS and indoor positioning can be made and by this integration positioning signal can be obtained everywhere (ubiquitous).

11. REFERENCES

[1] Molin, G., Pettersson, C., Jonsson, O. and Keijer, U., "Living at Home with Acquired Cognitive Impairment – Can Assistive Technology Help? Technology and Disability", IOS Press, pp 91-101, 2007.

[2] John Krumm, "Ubiquitous Computing Fundamentals", ISBN 978-1-4200-9360-5, September 21, 2009.

[3] Ward, A., Jones, A., and Hopper, A. "A new location technique for the active office", IEEE Personal Communications 4(5), 1997.

[4] Ming-Hui Jin, Chung-Jung Fu, Chih-Hao Yu, Hung-Ren Lai, and Ming- Whei, "I3BM ZigBee Positioning Method for Smart Home Applications," International Journal of Smart Home, Vol. 2, No.2, pp. 127-139, April, 2008.

[5] Janire Larranaga, Leire Muguira, Juan-Manuel Lopez-Garde, and Juan-Ignacio Vazquez, "An Environment Adaptive ZigBee-Based Indoor Positioning Algorithm," IEEE International Conference on Indoor Positioning and Indoor Navigation, Switzerland, pp. 1-8, 15-17 September, 2010.

[6] Miguel Dom´ınguez-Dur´an, D´ebora Claros, Cristina Urdiales, Francisco Coslado, and Francisco Sandoval, "Dynamic Calibration and Zero Configuration Positioning System for WSN," The 14th IEEE Mediterranean Electrotechnical Conference, MELECON, pp. 145-150, 5-7 May, 2008.

[7] Ambili Thottam Parameswaran, Mohammad Iftekhar Husain, and Shambhu Upadhyaya, "Is RSSI a Reliable Parameter in Sensor Localization Algorithms – An

Experimental Study," Department of Computer Science and Engineering, State University of New York at Buffalo, USA, pp. 1-5, 2009.

[8] Y. X. Zhao, Q. Shen, and L. M. Zhang, "A Novel High Accuracy Indoor Positioning System Based on Wireless LANs," Electromagnetics Research C, Vol. 24, pp. 25-42, 2011.

[9] J. Hightower, R. Want and G. Borriello; SpotON: An indoor 3D location sensing technology based on RF signal strength, UW CSE00-02-02, February 2000, http://www.cs.washington.edu/homes/jeffro/pubs/hightower2000indoor/hightower2000indoor.pdf

[10] RadioFrequencyIdentification (RFID) home page, http://www.aimglobal.org/technologies/rfid/, February 2012

[11] http://neuroph.sourceforge.net/, February 2012

[12] http://arduino.cc/en/Main/ArduinoBoardBT?from=Main.ArduinoBoardBluetooth, February 2012

[13] G. Zhang, S. Krishnan, F. Chin, and C.C. Ko., "UWB multicell indoor localization experiment system with adaptive TDOA combination", In Vehicular Technology Conference, IEEE 68th, pages 1-5, 2008.

[14] S. Gezici, Zhi T., G.B. Giannakis, H. Kobayashi, A.F. Molisch, H.V. Poor, and Z. Sahinoglu. "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks", Signal Processing Magazine, IEEE, 22(4):70-84, 2005.