Pervasive and Mobile Computing 8 (2012) 271–281

Contents lists available at SciVerse ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc

# Design and implementation of a self-guided indoor robot based on a two-tier localization architecture

Lun-Wu Yeh [a,*], Ming-Hsiu Hsu [a], Hong-Ying Huang [a], Yu-Chee Tseng [a,b]

[a] Department of Computer Science, National Chiao Tung University, Hsin-Chu, 300, Taiwan

[b] Research Center for Information Technology Innovation, Academia Sinica, Taipei, 115, Taiwan

ARTICLE INFO

*Article history:*
Received 3 August 2010
Received in revised form 15 February 2011
Accepted 7 April 2011
Available online 21 June 2011

*Keywords:*
Intelligent home
Localization
Pervasive computing
RFID
Robot

ABSTRACT

We consider building an indoor low-cost mobile robot that can be used in home applications. Due to the complicated nature of home environments, it is essential for such a robot to be self-guided in the sense that it is able to determine its current location as well as navigate to locations where it is commanded to. We propose a two-tier architecture to achieve this goal at centimeter-to-meter-level accuracy. The robot can even roam into an area which is new to it. We demonstrate a prototyping system based on an extended iRobot and the results have important implications on intelligent homes.



## 1. Introduction

Indoor mobile robots have lots of potential applications in daily life, such as home care, room cleaning, object finding, and emergency supporting [1–3]. Most mobile services would require a robot to determine its current location [4–9]. While this issue has been intensively studied in factory environments [5,10], it is more challenging for homes/offices, where the environment is not so strictly controlled and the interior planning is subject to change at any time. The purpose of this study is to design a suitable indoor navigation mechanism for mobile robots to meet this need.

To precisely navigate a robot, a localization method is needed [11–13]. GPS dominates many outdoor localization applications. However, for indoor localization, a globally usable solution is still missing. In Section 2, we will review some existing indoor localization solutions for mobile robots. In this work, we propose a framework to design a "self-guided" indoor robot. Our goal is to take advantage of existing technologies, such as RFID, wireless networks, and indoor localization systems to achieve this goal. We list some design guidelines of our framework:

- Plug-and-play. It is desirable that the robot can automatically start without going through any setup effort. For example, some robotic systems [14,15] need a training phase before getting started, but some [7–9] need not. The robots in Refs. [7–9] do not require to have any pre-knowledge about the environment before entering the environment.
- Self-guided. The robot should be able to guide itself to any location that it desires to and can still recalibrate itself. For example, the robots in Refs. [4–6] need not know their initial locations when entering an environment.
- Multi-level localization accuracy. Depending on different situations, different levels of accuracy, perhaps at different costs, can be provided. An example is the work [16].

* Corresponding author.
*E-mail addresses:* lwyeh@cs.nctu.edu.tw (L.-W. Yeh), jameshsu75@gmail.com (M.-H. Hsu), hungying@cs.nctu.edu.tw (H.-Y. Huang), yctseng@cs.nctu.edu.tw (Y.-C. Tseng).

1574-1192/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.pmcj.2011.04.008

- Low or no extra infrastructure. The extra infrastructures required to guide the robot should be as minimal as possible. Also, we may use existing infrastructures to achieve this goal.
- Low cost. We are not looking for complicated systems. The system should be inexpensive and affordable.
- Scalability. Our system should be scalable to larger environments without adding much overheads.

Following these guidelines, we propose a two-tier localization architecture for the design of a self-guided robot. Our system does not rely on building a new infrastructure. Instead, we utilize existing infrastructures, such as RFID and WiFi networks, which might be widely and densely deployed in future homes/offices. Assuming that the robot is equipped with a WiFi interface and an RFID reader, our system supports localization at two precision levels. A location server based on a WiFi fingerprinting method [11,12] helps the robot to position itself at meter-level precision. Also, on the ground, some RFID tags are deployed and a tag map is pre-installed in the location server to help the robot to position itself at centimeter-level precision. Through this two-tier mechanism, we can meet the above requirements at reasonable low cost and deployment overhead. The result should be applicable to many indoor navigation services in home/office environments. We also design a spiral search algorithm to help the robot to identify an RFID tag and analyze the corresponding cost (spiral moving distance) that an RFID tag is expected to be found.

The remainder of this paper is organized as follows. Section 2 gives some preliminaries. Section 3 presents our two-tier localization architecture. Our prototyping details and some performance evaluation results are presented in Section 4. Conclusions are drawn in Section 5.

## 2. Preliminaries

### 2.1. RFID and WiFi

RFID (Radio Frequency Identification) generally refers to systems that can transmit identity information of objects via wireless technologies. Such technologies have been widely applied to stock tracking, security, logistics, etc. A typical RFID system is composed of tags, readers, and servers. A tag is a tiny radio device with a simple silicon microchip attached to a small flat antenna mounted on a substrate. RFID tags have two types: *active* and *passive*. An active tag has its own power source and thus has a longer transmission distance. A passive tag, on the contrary, has no power source and thus can only respond to a reader when accumulating sufficient energy in its capacitor. Readers can send and receive RF signal to and from tags. Typical operation frequencies include 455 MHz, 2.45, and 5.8 GHz. Servers can be general computers to interact with readers to provide various applications.

The term WiFi stands for Wireless Fidelity. WiFi is a class of wireless local area network (WLANs) based on IEEE 802.11 standards [17]. Nowadays, it is widely used for broadband Internet access. A WiFi network consists of some access points (APs) and WiFi interfaces at clients. Recently, WiFi APs have been intensively studied for providing indoor localization when GPS signal is not available. In this work, we will apply such techniques for both guiding our robot and providing "RFID tag maps" on request of the robot.

### 2.2. Localization techniques

Localization solutions can be classified as AoA-based [18], ToA-based [19], TDoA-based [20], infrared-based [21], ultrasonic-based [22], RF-based [11], and fingerprint-based [11–13]. In this work, observing that WiFi networks are widely deployed, we are more interested in applying fingerprinting localization methods [11,12], which rely on existing WiFi APs as beacons. Such solutions have two phases. In the *training phase*, radio signal strength (RSS) patterns are collected at a set of training locations from pre-deployed beacons and stored in a database (called radio map). During the *positioning phase*, a device to be localized can collect its current RSS pattern and compare it against the radio map established in the training phase to identify its possible location. Several variants of fingerprinting techniques have been proposed [12].

### 2.3. Robot navigation techniques

For most mobile robot applications, it is essential to precisely locate and navigate a robot. Robot navigation techniques can be categorized as laser-based, track-based, and vision-based. In Ref. [4], three or more laser guiding beams are used to determine a robot's location. A track-based system uses some traces, such as magnetic tapes on the ground, to guide a robot [5]. Vision-based systems [6] are more flexible, but they require high computing resources and are constrained by line-of-sight.

Recently, RFID techniques have also been applied to robot navigation [7–9]. In Refs. [7,9], an array of passive RFID tags are deployed on the entire field. Each tag represents a unique location. On scanning any tag, the robot can calculate its location. Grid deployment is discussed in Ref. [7], while triangular deployment is discussed in Ref. [9]. Such systems normally require a large number of RFID tags, especially when the field is large. The work [8] uses a "rotatable" RFID reader to guide a robot to a stationary target with an active tag. An AoA-like model is adopted. This solution is more costly and is mainly for one single target. In comparison, our system uses sparse passive tags for centimeter-level positioning and it takes advantage of existing WiFi networks for meter-level positioning. Beside using RFID tags, the work [23] proposes a hybrid technology

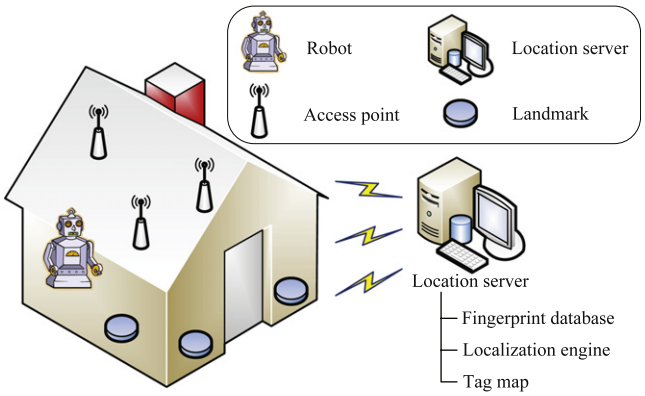


**Fig. 1.** The system architecture of our self-guided robot system.

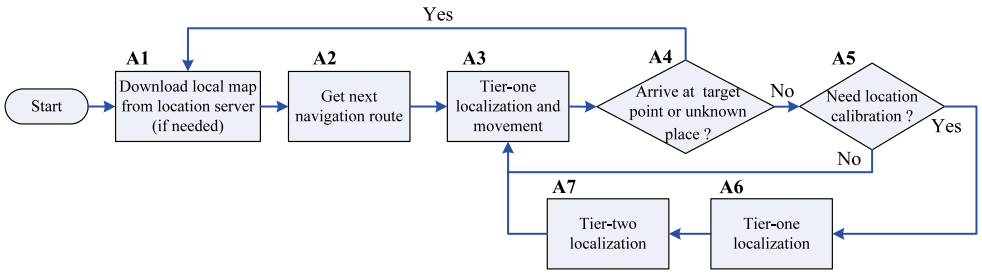


**Fig. 2.** Flow chart of our navigation procedure.

which combines an RFID system and a wireless sensor network. However, sensor nodes need to be deployed into a grid manner and the proposed algorithm is more suitable for open space with no interferences.

## 3. Design of a two-tier self-guided robot

### 3.1. System architecture

Fig. 1 shows the system architecture of our self-guided robot system. There are four main components:

- WiFi networks. The networks contain a number of access points, which periodically transmit beacons. Through these beacons (as reviewed in Section 2), a device can estimate its location at a meter-level accuracy through a fingerprinting mechanism.
- Location server. The location server runs a fingerprinting localization algorithm, such as those in Refs. [11,12]. It has three modules: (i) fingerprint database, (ii) localization engine, and (iii) tag map. The fingerprint database contains a set of training data, each binding a physical location in the target field with a vector of RSS patterns that can be received at that location. When localization is needed, the robot can collect its current RSS pattern and inquire the localization engine to determine its location by comparing to the fingerprint database (which we call meter-level positioning). On getting its location, the robot can further inquire the local tag map, which contains the locations of the landmarks deployed nearby the robot.
- Landmarks. These landmarks are passive, each of which has been registered in the tag map to identify a unique location. Finding any landmark is also considered a localization event (which we call centimeter-level positioning). These landmarks are divided into groups and each group of landmarks are deployed in a special shape (to be elaborated later on) to facilitate the robot to discover them. These tag groups may be deployed in a regular manner and at main turning points or service points.
- Robot. The robot is a mobile platform equipped with an e-compass, a WiFi interface, and an RFID reader. It conducts our two-tier localization scheme to determine its current location. In the first tier, it only needs to inquire the location server. In the second tier, it needs to search the tag map for landmarks and we will propose our searching strategy later on. The e-compass helps indicate the orientation of the robot. As to service provisioning, it may be equipped with other tools, such as cameras, arms, etc. (this is out of the scope of this work).

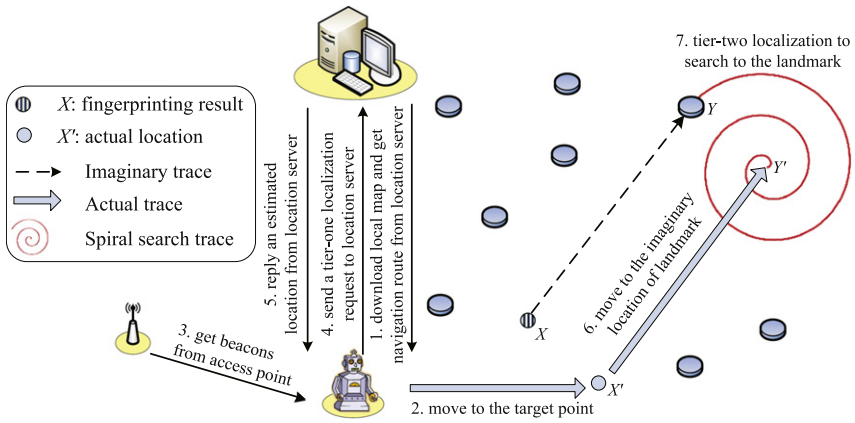


**Fig. 3.** An example of our two-tier localization.

### 3.2. Two-tier localization scheme

Fig. 2 shows the flow chart of our two-tier localization scheme for the robot. One fact for most motor systems is that a robot is subject to a certain degree of errors when it moves a certain amount of time or distance. Furthermore, moving on different types of floor materials (which is quite common in home/office environments) may also incur different degrees of errors. Below, we describe the detail procedures, which take the above errors into account.

A1. To initialize, the robot will download the local map from the location server (this can be done through the WiFi interface). Since we target at indoor applications, the map should include the floor plan and the interior design. Whenever the robot finds itself entering a new environment (e.g., by finding new WiFi APs), it can download a new map.

A2. After getting the local map, the robot will get the next navigation route and command from the location server.

A3. The robot then tries to move to the next target point. Depending on the tolerable error of the robot's motor system, the robot will enter A4 as necessary (e.g., it may need to calibrate its location after moving a certain distance or a certain amount of time). Also, it may enter A4 if it suddenly finds itself in an unfamiliar environment (e.g., it may find some unfamiliar WiFi APs).

A4. The robot checks if it has arrived at the target point or an unfamiliar environment. If so, it goes to A1; otherwise, it goes to A5. (Note that the check is application-dependent. It may be triggered, for example, by landing at a special landmark, contacting a charger, receiving a special signal, etc.).

A5. The robot then checks if it needs to calibrate its current location. If so, it goes to A6; otherwise, it goes to A3. (Note that the check may be done by conducting a tier-one localization. If the result deviates from its expected location by a certain threshold or above, a calibration is needed.)

A6. In this tier-one localization, the robot collects its current RSS patterns and inquires its location with the location server. Based on the result, it also asks for a local tag map from the location server.

A7. In this tier-two localization, the robot tries to scan nearby landmarks. Here, we propose to use a spiral search scheme (see Section 3.3). On tracking any landmark, the robot knows its accurate location. Then, it calibrates its orientation by its e-compass. Then, go to A3.

Fig. 3 shows an example of the above procedure. In step 1, the robot will download the local map and tag map and get the navigation route from the location server. In step 2, it tries to move to the target point. When the robot thinks that it has arrived at the expected target point, it checks if it needs to recalibrate itself. In step 3, it collects gets beacons from nearby access points and sends a tier-one localization request to the location server. In steps 4 and 5, it gets an estimated location $X$ at a meter-level precision from the location server. Since the fingerprinting localization is subject to error, let us suppose that its actual location is $X'$. In step 6, the robot starts to move to the landmark $Y$. Its actual trace is shown by the bold arrow, while it thinks that its trace is the dotted arrow. At point $Y'$, the robot is unable to find the landmark $Y$. Hence, it starts a tier-two localization using a spiral search to find the landmark $Y$ in step 7. Once the landmark is found, the robot positions itself at centimeter-level precision.

Also note that in A3, the robot may not need to move in a straight line in order to get to the next target point. To avoid accumulating too much error while moving, it may choose to land on some intermediate landmarks before reaching the target point. The problem of inserting appropriate intermediate landmarks can be done by a geographic forwarding algorithm, such as those in Refs. [24,25], where one can continuously choose the next landmark which is within a tolerable distance from its current location and is nearest to the target point. Fig. 4 shows an example, where landmarks A and B are selected as intermediate points.

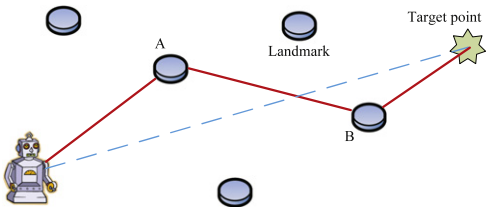


**Fig. 4.** An example of choosing intermediate landmarks.

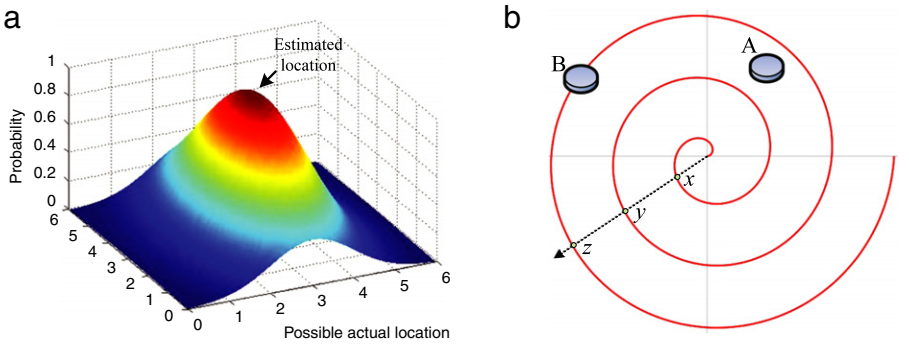


**Fig. 5.** (a) Typical probability distribution of the actual location obtained by the fingerprinting localization. (b) An example of spiral search.

### 3.3. Spiral search scheme

As mentioned above, our robot may know its approximated location as contributed by the tier-one localization result. However, if higher accuracy is needed, finding an RFID tag on the ground is needed. Therefore, in our tier-two localization, a search mechanism is needed to find a target landmark in step A7. We propose a heuristic scheme which uses the spiral search because it can gradually expand its search range. Searching for a target in an unknown environment has been discussed in Ref. [26]. The authors adopt the logarithmic spiral [27] to search the environment and show that it has good worst-case performance. On the contrary, our tier-one localization already provides meter-level precision of the robot's current location and a local tag map can be provided. There could be some difference between what the robot thinks where it is and where it is actually located. We propose to adopt the Archimedean spiral [28], which has successive turns at a constant separation, as illustrated in Fig. 5(b), to search the robot's surrounding. The problem is modeled as follows. After the fingerprinting localization, the PDF of the actual location of the robot with respect to the estimated location is formulated by a bell-shape 2D Gaussian distribution, as illustrated in Fig. 5(a). Then, we propose to adopt the Archimedean spiral to search robot's surrounding.

Without loss of generality, let the current location of the robot at the beginning of the search be the origin. The Archimedean spiral can be modeled by the polar coordinate $r = \mathrm{d}\theta$, where d is a constant. It is not hard to see that for any vector leaving from the origin, any two consecutive intersecting points of the vector and the spiral have a distance of $2\pi d$ (for example, the distance between points $x$ and $y$ in Fig. 5(b) is $2\pi d$, and the same for that between $y$ and $z$). As to be shown later, we will deploy RFID tags in groups, and each group has a certain shape that guarantees overlapping with any two such consecutive intersecting points (such as $x$ and $y$).

After leaving the spiral origin, the robot will try to scan any RFID on the ground (whenever a tag is found, the tier-two localization succeeds). In the example of Fig. 5(b), the robot will miss the RFID tag $A$, but will find the tag $B$ (note that successfully finding a tag is constrained by the sensing distance of the RFID reader). Because tags are quite cheap, to facilitate the search process, we propose to deploy RFID tags in groups, each of a certain shape. In Fig. 6(a), we show a cross-shape tag group. In Fig. 6(b), we show several other shapes of tag groups (circle, cross, and triangle). Along the circumference of a shape, we place RFID tags uniformly, each separated by a fixed distance. Clearly, different shapes will require different numbers of RFID tags (and thus different costs).

No matter which shape is used, we need to guarantee that for any orientation of a tag group, it will intersect with the Archimedean spiral. Let $r = \mathrm{d}\theta$ be the spiral. Recall that the spiral line is separated by a distance of $2\pi d$ in any location. Let us use a circle tag group with a diameter $2\pi d$ as a basis. Since its perimeter is $2\pi^2 d$, using the same perimeter, Fig. 6(b) shows the side lengths of different tag group shapes ($\pi^2 d$, $2\pi^2 d/3$, and $\pi^2 d/2$ for cross, equilateral triangle, and square, respectively). Unfortunately, as shown in Fig. 6(c), only the circle shape guarantees the intersection property; the other shapes $B$, $C$, and $D$ all fail to guarantee this property at certain orientations. In particular, we note that the cross-shape $B$

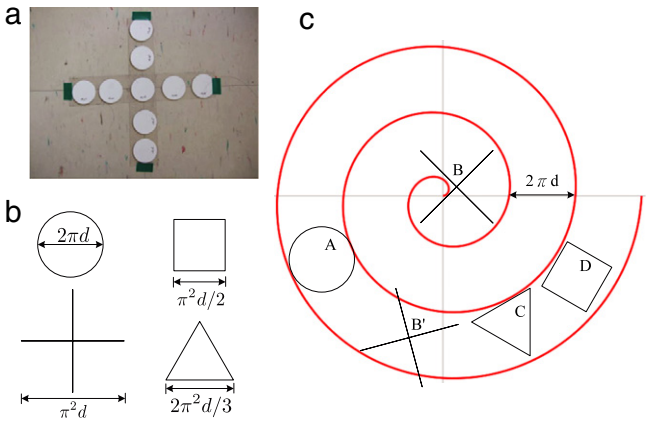


**Fig. 6.** Different shapes of RFID tag groups and their intersection properties.

only fails to intersect with the spiral nearby the origin of the spiral; after leaving the origin, the intersection property can be guaranteed at all orientations (see $B'$ in the figure).

It would be interesting to calculate the expected time that the robot can locate itself after starting a search. Assuming that the robot moves at a constant velocity and the error of the tier-one localization follows a 2D Gaussian distribution, we derive the following results.

**Lemma 1.** *Given an Archimedean spiral $r = \mathrm{d}\theta$ and a circle of radius $E$ centered at the origin, they intersect at only one point and the length of the spiral from the origin to intersection point is*

$$len\left(\frac{E}{d}\right) = \frac{1}{2}d\left[\frac{E}{d}\sqrt{1+\frac{E^2}{d^2}} + \ln\left(\frac{E}{d}+\sqrt{1+\frac{E^2}{d^2}}\right)\right].$$

**Proof.** Given any $\theta$ value, the arc length of the Archimedean spiral from the original to point $(r, \theta)$ is

$$len(\theta) = \frac{1}{2}d[\theta\sqrt{1+\theta^2} + \ln(\theta+\sqrt{1+\theta^2})].$$

Since $E = r = \mathrm{d}\theta$, replacing $\theta$ by $E/d$, we have proved this lemma. □

From Lemma 1, the expected value of our tier-two localization can be decided.

**Theorem 1.** *Assuming that the error of the tier-one localization follows a $(0, \sigma^2)$ Gaussian distribution such that the probability of the error distance $E$ is $f(E) = (1/\sqrt{2\pi\sigma^2})\exp(-E^2/2\sigma^2)$, the expected search distance of tier-two localization is*

$$\int_0^\infty len\left(\frac{E}{d}\right) f(E)\mathrm{d}E.$$

Note that the above spiral search has assumed that there is no obstacle that blocks the search path; otherwise, the search may fail. A route-around mechanism is needed when this happens.

## 4. Prototype and performance evaluation

At the Eco-City Lab [29] in the National Chiao Tung University, we have developed a prototype of the proposed self-guided robot. In this section, we report our prototyping experiences and some performance evaluation results.

### 4.1. Hardware and software components and design issues

Fig. 7 shows the components of the self-guided robot. We adopt iRobot Create [30] as the mobile platform and attach an RFID reader, an electronic compass, and a WiFi interface to the iRobot. The iRobot Create is a programmable mobile platform with flexibility for extension. It has a cargo bay with a 25-pin port that can accept digital and analog input and output. We connect an RS232 multi-port controller to the cargo bay. The RFID reader operates under 13.56 MHz and has a reading range of about 5 cm. The iRobot, RFID reader, and electronic compass are connected by the RS232 multi-port controller, through which the robot can get its (tier-two) location and orientation information. For convenience, we adopt the cross-shape

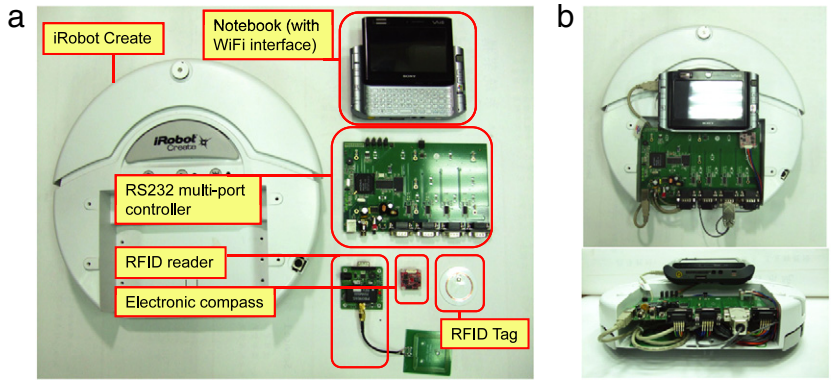


**Fig. 7.** (a) Hardware components of our self-guided robot. (b) Top and side views of the self-guided robot.

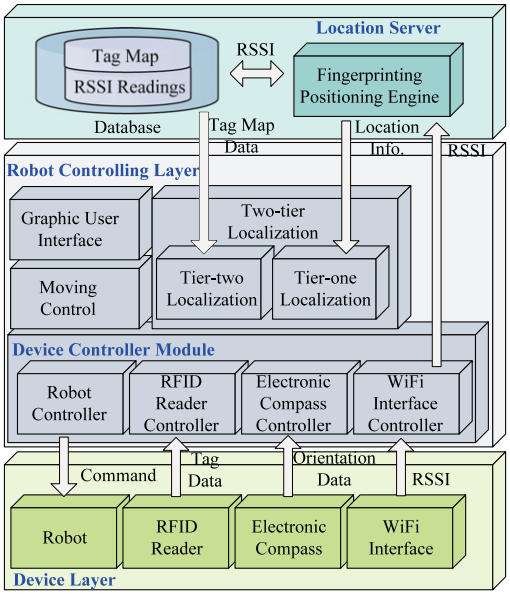


**Fig. 8.** The functional blocks of implementation.

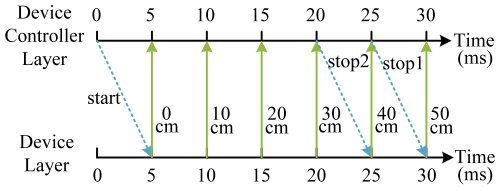


**Fig. 9.** An example of communication delay.

(30 cm × 30 cm) tag group as shown in Fig. 6(a) in our implementation. Each tag contains a unique ID and its coordinate is pre-registered to the system.

Fig. 8 shows the functional blocks of our implementation. The implementation architecture can be divided into three parts: *Device Layer, Robot Controlling Layer*, and *Location Server*. The Device Controller Module can get data from different devices respectively. The Robot Controlling Layer implements our two-tier localization scheme and the moving control mechanism. The Location Server contains a database, which stores a set of RSSI readings and RFID tag maps for tier-one and tier-two localization, respectively.

There are several implementation issues, which are discussed below.

1. Communication delay. The mechanism to get data from the robot is by polling. Initially, we send a request to start the operation. Devices will report periodically. The highest speed to send requests is 19,200 bps, thus causing a communication delay of 500–600 ms. This delay cannot be neglected when high accuracy is required. Hence, before

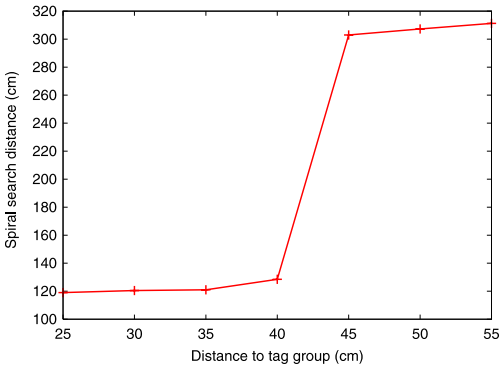


**Fig. 10.** Spiral search distance vs. initial distance from robot to tag group.

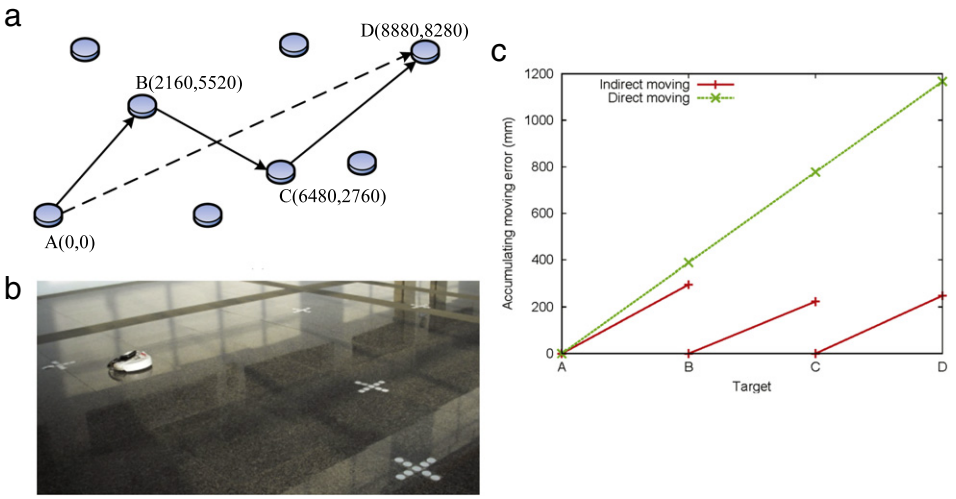


**Fig. 11.** The long distance moving experiment of self-guided robot.

running this system, we will measure the delay of each operation. Fig. 9 shows an example of controlling the robot movement. The communication delay of sending command is measured to be 5 ms. The Device Layer periodically reports the robot's moving distance to the Device Controller Module every 5 ms. Now, if we want to stop the robot at 40 cm when we hear such a report (i.e., dash line stop1), due to communication delay, the robot will stop at 50 cm. Therefore, we have to take the delay into account and send the stop command in advance at 20 ms (i.e., dash line stop2) to force the robot to stop at 40 cm.

2. Loading of wheels. The driving wheels of iRobot is by coaxial gear. If we put heavy objects on the robot, it will cause significant moving errors after moving a long distance. For example, if we add 1 kg to the robot, it will accumulate 1 m of error after moving 20 m. This also proves why we need the tier-two localization mechanism.
3. Response time of the RFID reader. When the robot reaches an RFID tag, the RFID reader will need to negotiate with the tags to get its information. The negotiation time is proportional to the data length in the tag. Hence, sometimes we need to control the moving speed of the robot to avoid negotiation failure.

### *4.2. Evaluations*

In this section, we evaluate our self-guided robot by three experiments. At first, we evaluate the efficiency of spiral search scheme. Then, we evaluate the two-tier localization scheme for long distance moving and moving around in our experimental environment.

To understand the performance of our spiral search, we deploy a cross-shape tag group of size 29 cm $\times$ 29 cm and we place the robot at different distances from the tag group. Then, we instruct the robot to search the tag group by spiral search until a tag is found. As shown in Fig. 10, when the distance to the tag group is less than 40 cm, the robot only needs to move no more than one spiral to find a tag, thus causing a search distance about 120 cm. When the distance to the tag group is larger than 40 cm, the robot will need to move two spirals, thus causing a quick jump to 300 cm to find a tag.

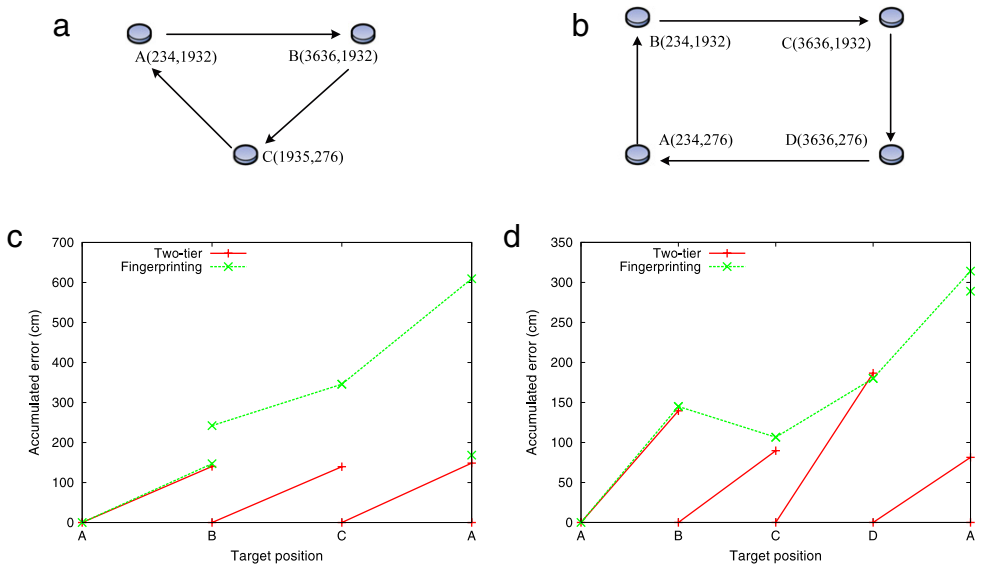


**Fig. 12.** Comparison of localization errors.

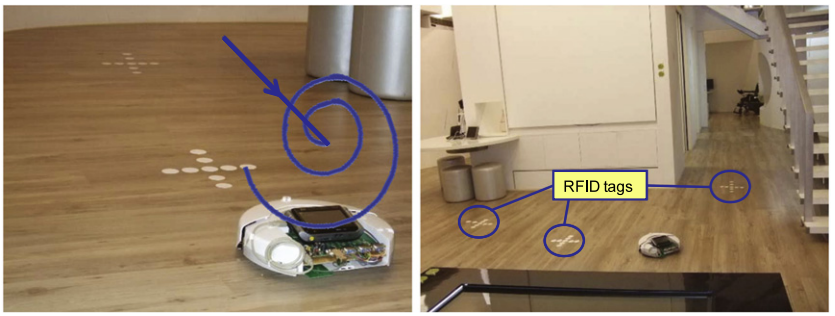


**Fig. 13.** An implementation scenario of our self-guided robot for automatic radio map construction.

Next, Section 3.2 mentions that we will insert some intermediate points while the robot moves long distance. As shown in Fig. 11(a) and (b), we place several RFID tags on the ground and the number in parentheses are the location of the RFID tag. We compare the robot moves from point *A* to point *D*, denoted as direct moving, and from *A* to *B*, *C*, and *D*, denoted as indirect moving. Fig. 11(c) shows that indirect moving can avoid larger accumulated error than direct moving. Hence, if we apply the self-guided robot to the high accuracy application, we should insert more intermediate points to reduce the accumulated error.

To demonstrate why our two-tier localization scheme would work better than the typical WiFi fingerprinting localization [11], we send our robot to traverse the paths as shown in Fig. 12(a) and (b). At each turning point, we deploy an RFID tag group. For the fingerprinting algorithm, we conduct localization once at each turning point and measure its distance to the corresponding turning point (this distance is regarded as localization error). For our two-tier localization, we will conduct tier-one localization at each turning point and then do a tier-two spiral search. The comparison is shown in Fig. 12(c) and (d). It shows that error will not be accumulated in our scheme. Therefore our two-tier localization scheme is quite suitable for indoor robot applications.

## 5. Conclusions and further applications

Indoor robots have lots of potential applications in our daily life. Since indoor design could be quite complicated, a key issue is the self-localization capability of such robots. In this paper, we have proposed a two-tier localization architecture for indoor robots. Our self-guided robots rely on existing widespread WiFi and RFID technologies. We have also discussed the design of different RFID tag groups and the expected spiral searching distance. In addition to theoretical results, we have also implemented a prototype system and evaluated its performance. Our current robot platform can only move on smooth floors. How to move on bumpy floors or even from one floor to another deserves further study. In addition to indoor localization, our system may be applied to other scenarios in our daily life. Below, we present two case studies that we have conducted at the National Chiao Tung University.

1. Tour Guide. In large exhibition areas, exhibition contents are provided to visitors in a location-aware manner. Our platform can be naturally extended to such applications. We have explored such a possibility using landmark RFID tags, exhibition RFID tags, WiFi access points, location server, and self-guided robots in our campus. The exhibition RFID tags are to provide exhibition information. Interestingly, our self-guided robots cannot only guide visitors but also help plan visiting routes dynamically according to their preferences.
2. Automatic Radio Map Construction. For indoor localization, a globally usable solution is still missing. Here, we are interested in the fingerprinting-based solution, such as those in Refs. [11–13]. A major drawback of the fingerprinting approach is its labor-intensive training process, especially in a large-scale field. Further, since RF signal is inherently unstable, the radio map collected earlier may deviate significantly from the current one [31]. Manually calibrating radio maps is error-prone. Besides, the environment may change, furniture may be moved, and beacons may be reconfigured or upgraded anytime [32]. All these factors may result in non-negligible errors when positioning an object. To relieve these problems, we can send our self-guided robot to collect the current signal strength automatically whenever needed. The robot can thus automate the training process and frequently update radio maps to reflect the current RSS patterns. This not only significantly reduces human labors but also improves positioning accuracy. Fig. 13 shows our implementation scenario.

## Acknowledgements

Y.-C. Tseng's research is co-sponsored by MoE ATU Plan, by NSC grants 97-3114-E-009-001, 97-2221-E-009-142-MY3, 98-2219-E-009-019, and 98-2219-E-009-005, 99-2218-E-009-005, by ITRI, Taiwan, by III, Taiwan, by D-Link, and by Intel.

## References

[1] K.-T. Song, C.-Y. Tsai, F.-S. Huang, J.-W. Hong, C.-Y. Lin, C.-W. Chen, Z.-S. Lin, Development of the robot of living aid: rola, in: Proc. of IEEE Int'l Conference on Automation and Logistics, 2008.
[2] R. Barea, L. Bergasa, E. Lopez, M. Ocana, D. Schleicher, A. Leon, Robotic assistants for health care, in: Proc. of IEEE Int'l Conference on Robotics and Biomimetics, 2008.
[3] T. Taipalus, K. Kosuge, Development of service robot for fetching objects in home environment, in: Proc. of IEEE Int'l Symposium on Computational Intelligence in Robotics and Automation, 2005.
[4] K. Lingemanna, A. Nuchtera, J. Hertzberga, H. Surmannb, High-speed laser localization for mobile robots, Robotics and Autonomous Systems 51 (2005) 275–296.
[5] Automated guided vehicle, 2010. http://en.wikipedia.org/wiki/Automated_Guided_Vehicle.
[6] E.M. Lowa, I.R. Manchesterb, A.V. Savkina, A biologically inspired method for vision-based docking of wheeled mobile robots, Robotics and Autonomous Systems 55 (2007) 769–784.
[7] B.-S. Choi, J.-W. Lee, J.-J. Lee, An improved localization system with RFID technology for a mobile robot, in: Proc. of conference of IEEE Industrial Electronics, IECON, 2008.
[8] M. Kim, N.Y. Chong, RFID-based mobile robot guidance to a stationary target, in: Mechatronics, 2007.
[9] H. Lim, B. Choi, J. Lee, An efficient localization algorithm for mobile robots based on RFID system, in: Proc. of Int'l Joint Conference on SICE-ICASE, 2006.
[10] L. Shengfang, H. Xingzhe, Research on the agv based robot system used in substation inspection, in: Proc. of Int'l Conference on Power System Technology, 2006.
[11] P. Bahl, V.N. Padmanabhan, Radar: an in-building RF-based user location and tracking system, in: Proc. of IEEE INFOCOM, 2000.
[12] S.-P. Kuo, Y.-C. Tseng, A scrambling method for fingerprint positioning based on temporal diversity and spatial dependency, IEEE Transactions on Knowledge and Data Engineering 20 (2008) 678–684.
[13] J.J. Pan, J.T. Kwok, Q. Yang, Y. Chen, Multidimensional vector regression for accurate and low-cost location estimation in pervasive computing, IEEE Transactions on Knowledge and Data Engineering 18 (2006) 1181–1193.
[14] V. Ganapathy, S.C. Yun, J. Ng, Fuzzy and neural controllers for acute obstacle avoidance in mobile robot navigation, in: Proc. of IEEE/ASME Int'l Conference on Advanced Intelligent Mechatronics, 2009.
[15] V. Ganapathy, S.C. Yun, H. Joe, Neural q-learning controller for mobile robot, in: Proc. of IEEE/ASME Int'l Conference on Advanced Intelligent Mechatronics, 2009.
[16] J.Y. Park, H.Y. Song, Multilevel localization for mobile sensor network platforms, in: Proc. of Int'l Multiconference on Computer Science and Information Technology, 2008.
[17] IEEE Std 802.11, IEEE standard for information technology-telecommunications and information exchange between systems – local and metropolitan area networks-specific requirements – part 11: wireless lan medium access control (MAC) and physical layer (PHY) specifications, 2007.
[18] D. Niculescu, B. Nath, Ad hoc positioning system (aps) using aoa, in: Proc. of IEEE INFOCOM, 2003.
[19] M. Addlesee, R. Curwen, S. Hodges, J. Newman, P. Steggles, A. Ward, A. Hopper, Implementing a sentient computing system, IEEE Computer 34 (2001) 50–56.
[20] A. Savvides, C.-C. Han, M.B. Strivastava, Dynamic fine grained localization in ad-hoc networks of sensors, in: Proc. of ACM Int'l Conference on Mobile Computing and Networking, MobiCom, 2001.
[21] E. Brassart, C. Pegard, M. Mouaddib, Localization using infrared beacons, Robotica 18 (2000) 153–161.
[22] N.B Priyantha, A. Chakraborty, H. Balakrishnan, The cricket location-support system, in: Proc. of ACM Int'l Conference on Mobile Computing and Networking, MobiCom, 2000.
[23] D. Zhang, Y. Yang, D. Cheng, S. Liu, L.M. Ni, Cocktail: an rf-based hybrid approach for indoor localization, in: Proc. of IEEE Int'l Conference on Communications, ICC, 2010.
[24] B. Karp, H.T. Kung, Gpsr: greedy perimeter stateless routing for wireless networks, in: Proc. of ACM Int'l Conference on Mobile Computing and Networking, MobiCom, 2000.
[25] M. Witt, V. Turau, Bgr: blind geographic routing for sensor networks, in: Proceedings of the Third Workshop on Intelligent Solutions in Embedded Systems, 2005.
[26] S. Burlington, G. Dudek, Spiral Search as an Efficient Mobile Robotic Search Technique, Technical Report, 1999.
[27] Logarithmic spiral, 2010. http://en.wikipedia.org/wiki/Logarithmic_spiral.
[28] Archimedean spiral, 2010. http://en.wikipedia.org/wiki/Archimedean_spiral.
[29] Eco-City, 2008. http://www.ecocity.org.tw/.
[30] iRobot, 2010. http://store.irobot.com/corp/index.jsp.

[31] J. Yin, Q. Yang, L.M. Ni, Learning adaptive temporal radio maps for signal-strength-based location estimation, IEEE Transactions on Mobile Computing 7 (2008) 869–883.
[32] S.-P. Kuo, H.-J. Kuo, Y.-C. Tseng, The beacon movement detection problem in wireless sensor networks for localization applications, IEEE Transactions on Mobile Computing 8 (2009) 1326–1338.