Redpin - Adaptive, Zero-Configuration Indoor Localization through User Collaboration

Philipp Bolliger
Institute for Pervasive Computing
ETH Zurich, Switzerland
bolligph@inf.ethz.ch

ABSTRACT

Redpin is a fingerprint-based indoor localization system designed and built to run on mobile phones. The basic principles of our system are based on known systems like Place Lab or Radar. However, with Redpin it is possible to consider the signal-strength of GSM, Bluetooth, and WiFi access points on a mobile phone. Moreover, we devised methods to omit the time-consuming training phase and instead incorporate a folksonomy-like approach where the users train the system while using it. Finally, this approach also enables the system to expeditiously adapt to changes in the environment, caused for example by replaced access points.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Clustering, Implementation; C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*client/server*

General Terms

Measurement, Human Factors, Performance

Keywords

Positioning Systems, Indoor Localization, Zero-Configuration, User Collaboration, Adaptive Algorithms, Open-Source

1. INTRODUCTION

The location of a user or a device is a very meaningful and significant information for many applications in the field of ubiquitous computing [13, 27]. It certainly is the most prominent contribution when it comes to determining a user's context or activity [7]. While determining the position of a mobile device outdoors is accurately possible in many situations using GPS, there is no free and easy to use system for indoor localization as far as we know.

Still, research in the past few years has shown that fingerprinting is the most promising approach to determine the location of a mobile device in various indoor settings with very different signal propagation characteristics. Hence, a lot of research focused on solving the problems that arise when using the received signal strength (RSS) to fingerprint a location, such as detecting and modeling line-of-sight obstructions [24], absorption by humans, or reflection on walls. In addition, a lot of effort was spent on finding accurate and robust algorithms to select a known fingerprint given a current RSS measurement, for example [9, 16, 17, 21, 23] . We will cover related work more extensively below.

However, most currently known localization systems that apply fingerprinting work in two phases. An offline training phase and an online phase in which data from the first phase is used to determine the current position of a mobile device. Moreover, these systems are only as accurate as the training phase has been detailed.

In this paper, we present a novel approach to fingerprinting that does not require an explicit offline phase but rather incorporates the training of the system into the usage and thus makes it an ongoing process that allows it to quickly adapt to changes in the environment.

1.1 Related Work

As mentioned above, the current location of a user or even a device is a very useful information for many different applications. Thus a lot of research has been done to determine the location of mobile devices indoors. A few commercial systems have also been introduced to address this need.

One class of indoor localization systems that have been demonstrated to be very accurate are systems that use special hardware (for example, RFID [10], infrared [28], or ultrasound [11]). Although being very accurate, such systems usually require the installation of hardware that is needed for the localization.

Another approach is to use empirical measurements of radio signals emited by already installed hardware. Using this fingerprinting approach, systems have been proposed that use GSM signals (for example [26]), WiFi signals (for example [2, 16]), and also Bluetooth signals [4]. Extending this idea, LaMarca et al. [19] are using multiple wireless technologies simultaneously to increase both the robustness as well as the accuracy of localization. But most of these systems require a designated training phase that can take an extended period of time.

The work of Lim et al. [22] requires no training by automating the calibration of the effect of wireless physical characteristics on RSS measurements. But such automation requires very accurate RSS readings and thus the usage of sensitive WiFi network cards. With Redpin, however,

MELT'08, September 19, 2008, San Francisco, California, USA.

we want to use standard mobile phones which are usually equipped with smaller and hence less sensitive antennas.

The rest of this paper is organized as follows. In the next section, we will first motivate our system and conclude with our goals. In Section 3 we describe the basic mechanisms while exploring the architecture in detail in Section 4. In Section 5 we show how Redpin localizes mobile devices and present first results of our method in Section 6. We conclude this paper with our findings and a short outlook on future research directions.

2. MOTIVATION

Although Marc Weiser's vision of ubiquitous computing [29] is finally starting to become reality, one key missing technology that still hinders a broad emergence of such applications is a ubiquitously available localization system. Despite the fact that commercial systems like Ekahau[1] or UbiSense[2] are very accurate, the costs of installation, maintenance, and thus ownership are still high. Moreover, most commercial systems require one to purchase and install specific hardware, i.e., they can not be used with portable devices already at hand.

Academic systems that have been made publicly available like Place Lab [19] on the other hand are not easy to setup[3] and require one to train the system afterwards. In addition, the offline phase is usually time consuming, as these systems try to optimize the accuracy of the localization, which increases with the quality of the trained fingerprints. The COMPASS system for example is able to determine the position with an average error distance of less than 2.05 meters [16] using WiFi RSS. Yet, in order to reach this accuracy, it was necessary to measure at grid-aligned points with a spacing of only 1 meter and take measurements in 8 different directions at each point. Even in a very small building with a floor area of, for example, 125 m^2, the training phase would take more than 4 hours[4]. But the biggest issue with having a designated training phase is that is has to be repeated whenever the environment changes, for example due to a replaced access point.

However, such accuracy is only feasible when the measured signal strengths fluctuate only very slightly. Our own measurements (see Section 6) showed that the RSS of GSM signals can change up to 30% in only a few dozen seconds and the RSS of WiFi access points can even slip more than 50% within one hour. Furthermore, the RSS of WiFi access points depends heavily on whether humans are in the line of sight as the human body absorbs electromagnetic radiation quite well [18]. Hence, in rooms where the number of people is high and changes frequently, it seems unlikely that an accuracy of under 2 meters can be achieved.

Luckily, however, surveys of the most prominent challenges and issues in ubiquitous computing have shown that it is sufficient to localize a user within room-level accuracy for almost all applications [1, 3, 6, 12, 15, 30].

[1]http://www.ekahau.com
[2]http://www.ubisense.net
[3]It took one of our students almost two days to get the system running on just one mobile phone.
[4]This is, if we account 20 seconds per measurement, which is about the amount of time we experienced in our own experiments. See Section 6 for details.

2.1 Project Goals

Based on the findings presented above, we decided to build a fingerprinting-based indoor localization system. In addition we wanted our system to meet the following requirements:

- Use hardware that everyone already has
- Doesn't require the installation of special hardware in the building
- Must be very easy to setup and to maintain
- Should at least provide room-level (i.e., small office) accuracy
- Must be able to adapt to changes in the environment

One of our main goals with this project was to not only develop a system that would be easy to setup and use, but also to make it publicly available and release it under an open source license.

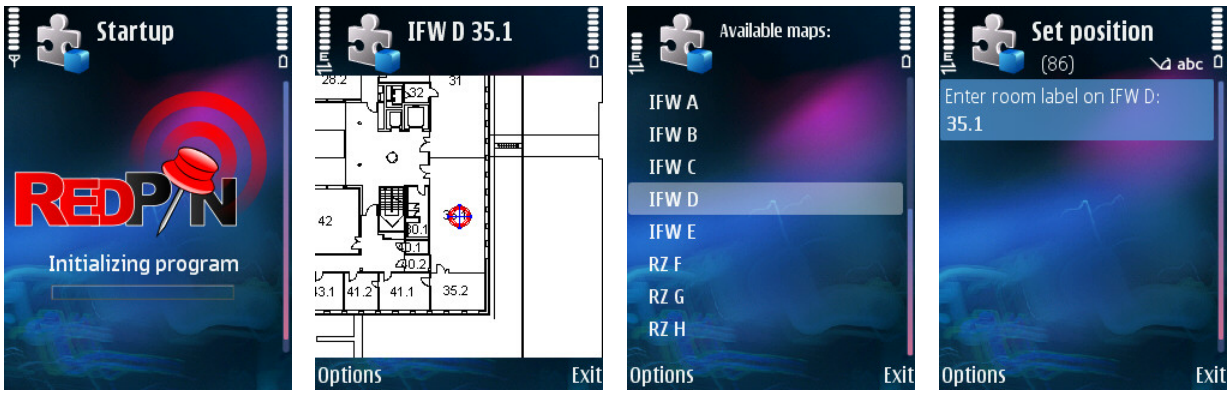
3. BUILDING PRINCIPLES

Given the requirements listed above, we decided to use symbolic identifiers to denote locations, thereby forgoing a potentially erroneous calculation of exact geographic coordinates. Consequently, localization of a mobile user or device can be reduced to the problem of mapping a set of RSS measurements to a known symbolic identifier, like for example a room number (see Figure 4). As mentioned in the introduction, this information is sufficient for most applications in ubiquitous computing. The unique assignment of a *measurement* to a *location* is called a *fingerprint*. Note however, that many fingerprints may be assigned to the same location.

In order to achieve room level accuracy, i.e., selecting the correct location given a measurement, we measure the signal strength of the currently active GSM cell, the signal strength of all WiFi access points as well as the Bluetooth identifier of all non-portable Bluetooth devices in range. We could additionally increase the system's accuracy by measuring the signal strength of all GSM cells, and not just the one GSM cell that is currently active, but due to technical limitations this is not currently possible with the devices we used (see Section 4 for details). But still, it is possible to determine the indoor location with a median localization error of 2.32 meters using only WiFi RSS [22], we were confident to be able to correctly map the combination of all readings to a location in almost every case. Moreover, reducing the problem to this simple mapping of room-level identifier entails the advantage that rooms are divided by walls that absorb and/or reflect electromagnetic radiation and thereby contribute in making a measurement unique.

3.1 User Collaboration

As mentioned in the introduction, we wanted our localization system to be easy to use and in particular easy to setup. First and foremost, our system should not require a designated training phase. Thus, after installing the software, we have no information about neither the building(s) nor the WiFi access points and Bluetooth devices installed therein.



(a) (b) (c) (d)

Figure 1: Using Redpin on a Nokia N95.

The key concept of Redpin is to let the users of the system create and manage the locations (i.e., the symbolic identifiers that denote a location) in a collaborative way. Using Redpin, every user can create, modify and, most importantly, use location information that was created by other users. We believe that this collaborative approach is feasible as people evidently like to participate and contribute to folksonomy-based systems. The massive success of websites such as Wikipedia[5] or OpenStreetMap[6] is just one evidence for this. Recent research in this area has in addition shown that people contribute because of ideological reasons and even more so, because it is fun [5, 25]. This however entails, that a system that relies on the contribution of its users should provide an appealing user interface.

In our first implementation, we do not actually capture the concept of a user, i.e., every mobile device that contributes to the system uses the same database of fingerprints. This allows to easily share knowledge about locations and enables a quick mapping of a building (see Section 6). On the other hand, this aspect entails security implications which are not yet addressed in our current work.

3.2 Redpin in Action

After installing Redpin on its mobile device, for example his or her mobile phone, the user can start-up the application right away. During initialization (see Figure 1(a)), the application is measuring the RSS of the active GSM cell, the RSS of all WiFi access points in range as well as the ID of all non-portable Bluetooth devices. We call this process *sniffing*. This measurement is then sent to a central server which will subsequently try to locate the mobile device given all known fingerprints. If the system can locate the mobile device, the user is presented the plan of the floor and his or her current location, indicated by a red, pulsing circle, as illustrated in Figure 1(b).

If the system can not locate the mobile device, for example because the location is yet unknown, Redpin will display the last known location. In the background, the system will continuously take measurements and compare the last three measurements, thereby trying to detect a *stable state* (see Section 5 for details). Upon detecting a stable state, the system will again try to locate the device. If the device can not be located, the user will be prompted to name the place of his or her current location and indicate the appropriate position on the floor plan. Thus, the user can choose from a list of known floor plans (see Figure 1(c)), set the marker (blue cross) to its current position, and enter the name of the current location, for example the room number as illustrated in Figure 1(d). In addition, a user can always correct the location in case Redpin provided the wrong identifier. This way, several fingerprints may be stored for the same identifier with a different timestamp. This mechanism allows Redpin to continuously adapt to changes in the electromagnetic environment.

In order to display not only the name of the current location but also show the position on the floor plan, the system must be given image files of each floor. These images can be uploaded to the server at anytime. However, the system does not require floor plan images since a location is defined solely by its symbolic identifier in Redpin.

4. ARCHITECTURE

Redpin consist of two basic components: a *Sniffer* component that gathers and collects information about different wireless devices in range in order to create a *fingerprint*, and a *Locator* component, which stores measured fingerprints in a repository and contains the algorithm to locate a mobile device. While the Sniffer component has to run on the mobile device for obvious reasons, the Locator component can be run either on a central server or on each mobile device separately. Although running the Locator, and hence storing the fingerprints, locally would be beneficial considering the users privacy, we need to store this data on a central server in order to allow for users collaborating. Thus, we implemented the Locator as a server, using Java SE 5.0 and MySQL as illustrated in Figure 2. We use Java Micro Edition for the GUI and all communication aspects, and Symbian Series 60 for the Sniffer component. This separation was necessary, as only the Symbian API would allow us to get the information we wanted to collect. For both the server and the mobile client, we used the Open Bandy[7] library to handle the serialization, transmission, and storage of the measured data.

[5]http://www.wikipedia.org
[6]http://www.openstreetmap.org
[7]http://www.openbandy.org

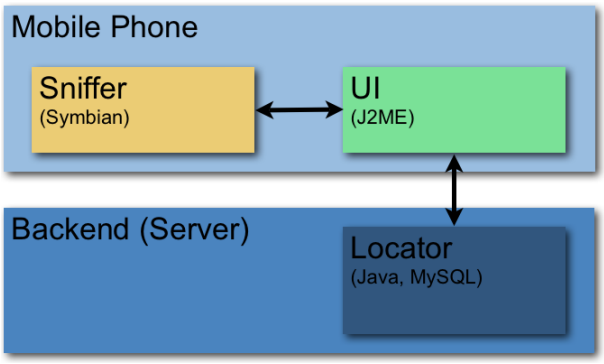


Figure 2: Redpin system architecture overview.

4.1 Server

The Redpin server provides several services for mobile clients. First of all, it provides a service that allows it to *store fingerprints* in a central database. This service is called whenever a mobile user stores or redefines a location. Another service allows the mobile clients to *retrieve maps*, i.e., images of the floor plan that are associated with a certain location. And most importantly, the server provides a service to *locate* a mobile device, i.e., to retrieve the fingerprint, and thus the location that best matches the measurement taken by the mobile client.

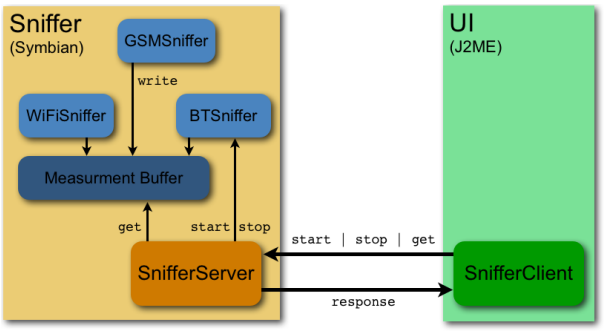


Figure 3: Sniffer architecture on the mobile phone.

4.2 Mobile Client

As mentioned above, our decision to use Symbian made it necessary to have two applications on the mobile device as illustrated in Figure 3. As we wanted our application to be as portable as possible, and in order to reuse code written for the server, we decided to implement the client software in Java ME. But as the limited API of Java ME would not allow access to the current RSS of neither the GSM nor WiFi, we had to implement the Sniffer component in Symbian. Hence, the Sniffer maintains a separate, asynchronous thread for each signal type (GSM, WiFi, and Bluetooth) that collects the appropriate information and stores it in a common buffer. This is necessary, as scanning GSM and WiFi signal usually is a matter of seconds whereas scanning for Bluetooth devices can take up to 2 minutes, depending on how many devices there currently are. To alleviate this problem, we additionally limit the Bluetooth scanner to 10 seconds. Eventually, the Sniffer communicates its current measurement to the Java MIDlet via a local TCP socket. The Java MIDlet on the other hand provides the user with the GUI depicted in Figure 1 and handles all the communication with the server.

5. POSITIONING

Because a location is simply expressed by a symbolic identifier in Redpin, the problem of calculating the current position is reduced to the problem of finding one fingerprint that best matches the given measurement. Hence, to determine the current location of a mobile device, we need to find the one known fingerprint that matches the current measurement best.

5.1 Sniffer Measurements

To increase the overall localization accuracy, in our case the success rate of calculating the correct location identifier, we measure three different signal sources, namely GSM, WiFi, and Bluetooth. In addition, we try to read the RSS of as many different sources as possible; a wide range of signal-strength fingerprints have been shown to increase accuracy of indoor localization systems significantly [26]. While both GSM and WiFi signals may fluctuate, Bluetooth devices are not always detected in the very short time range when we scan for devices. As a result, measurements can differ considerably, even when taken at the same place and in short succession. Hence, the biggest advantage of having *combined* fingerprints of GSM, WiFi, and Bluetooth signals is that we can adapt the localization algorithm dependent on the actual measurement.

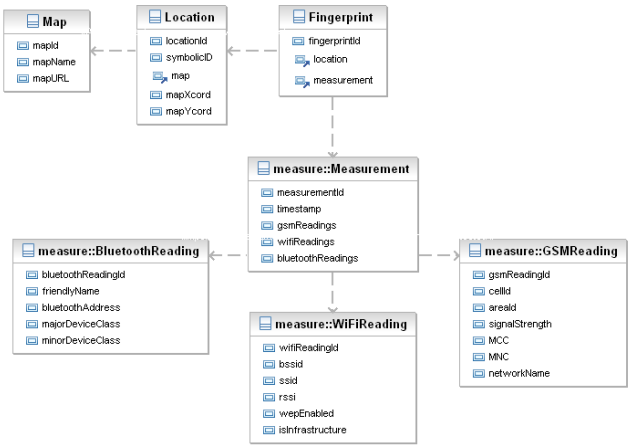


Figure 4: The Redpin data model.

To create an internationally unique GSM identifier, we readout the cell identifier (CI), the mobile country code (MCC), the mobile network code (MNC), as well as the location area code (LAC). In addition, we also retrieve the current received signal strengths (RSS) as an absolute value. Unfortunately, the current version of Symbian's Telephony API only provides information about the currently active cell. We hope to be able to read information about all GSM cells in range in a future version.

As the basic service set identification (BSSID) is unique by definition, it is sufficient to get this value along with the according RSS. Unlike with GSM, we are able to collect this information about all WiFi access points in range.

Bluetooth devices can be uniquely identified by the Bluetooth device address (BD_ADDR), similar to the MAC addresses of a network card. However, as we only want to consider non-portable devices, we have to retrieve the major and the minor device class during inquiry. This way, we can ignore mobile devices like mobile phones or portable audio devices that would distort the result otherwise. The RSS, although available on the Bluetooth host controller interface (HCI), is not exposed in the Symbian API[8].

5.2 Distance Measure

In order to compare different measurements, we defined a very simple distance measure for our first prototype. Note that the quality of this measure greatly accounts for the accuracy of the localization and we are currently working on more sophisticated algorithms and methods.

The *distance* between two measurements is computed using a straightforward account model. Every matching unique identifier, for example a WiFi BSSID that occurs in both measurements, adds to the cumulative distance while differing identifiers cause a diminution. For every matching pair of identifiers, an additional contribution is calculated based on the measured RSS, provided that the signal strength could be measured. This contribution can be negative in case the RSS values differ too much.

A *stable state* can thus be detected, by comparing the distance measure of at least three successive measurements. If the distance between all measurements is lower than the threshold, we assume that the mobile device has not been moved.

5.3 Locator Algorithm

To locate a mobile device, the Locator compares the current measurement with all known fingerprints in the database by calculating the distance measure as described above. If a fingerprint can be found whose distance to the current measurement is smaller than the threshold, the associated location will be returned to the mobile device. If multiple fingerprints are found, the system will return the best match.

6. EVALUATION

Given the requirements and goals defined for the Redpin system, we can evaluate the systems performance by answering two questions. First, how good is the localization, i.e., in how many cases is the room correctly determined? And second, how long does it take until a device can be located in every room, i.e., until the map for a building is complete?

To get answers to these questions, we installed the client software on multiple mobile phones[9] and conducted several experiments in our office building. In order to investigate the success rate, we added the fingerprints of 26 randomly chosen rooms to our database as illustrated in Figure 5. Note that some rooms in this building are smaller than 5 by 3 meters. Subsequently, we used another mobile device to determine the current location. We repeated the verification several times and over several days, during work hours as well as in the night. Overall, the system located the device in the correct room in 9 out of 10 cases. The cases where the algorithm returned the wrong identifier could be explained by our threshold settings, which we set to very strict values in order for the system to work in buildings with small rooms. In this case, the Locator would return the identifier of a room next to the one sought-after. Note that we never added additional fingerprints during the experiment to adapt to changes in the environment.

[8]Symbian Version 9.2

[9]We used two Nokia N95 and one Nokia N95 8GB for all our tests.

Figure 5: Points where measurements were taken. The labels A to W indicate on-floor measurements while X, Y, and Z indicate measurements that were taken on the stairs between the floors.

Given these results, the time it takes to get at least one fingerprint for every room depends only on how active users are in contributing to the system and on their mobility. A very short survey showed that when only 10 (out of 50 people working on this floor) contribute to the system, the map is complete after just one day.

7. CONCLUSIONS

In this paper we have presented Redpin, an adaptive indoor localization system that does not require a designated and time consuming training phase. The main contribution of our work is a novel approach to training fingerprint-based localization systems utilizing user contribution and hence allow collaboration. Initial experiments showed that even with a very simple distance measure and locator algorithm, the system achieves a success rate of about 90%. Moreover, we found that in most cases it is sufficient to have only one fingerprint per room. Thus, to get a complete map of an office building, only a few users must actually contribute to the system. Also, as Redpin allows several fingerprints for the same location, it is able to adapt to changes in the environment since users can always enter new fingerprints by correcting their location.

While results with the very simple locator algorithm presented are already viable, the next steps for the project are to explore and investigate more sophisticated ideas inspired by [8, 22]. We might also be able to automatically learn places and locations by applying the concepts described in [20] and in particular [14]. In addition, we are continuously testing our software on more mobile devices and hope to support different platforms soon.

As one of our goals was to make the system publicly available, Redpin is open source (LGPL) and available for download at www.redpin.org.

8. ACKNOWLEDGMENTS

The author would like to thank Diego Browarnik, Andreas Kamilaris, Davide Spena, and Simon Tobler for the implementation of the many parts that comprise Redpin. I would also like to thank Kurt Partridge and Norman Su for their thoughtful reviews and advice regarding draft versions of this paper.

9. REFERENCES

[1] G. Abowd and E. MYNATT. Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction*, Jan 2000.

[2] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. *INFOCOM 2000*, Feb 2004.

[3] M. Baldauf, S. Dustdar, and F. Rosenberg. A survey on context-aware systems. *International Journal of Ad Hoc and Ubiquitous Computing*, 2(4):263–277, 2007.

[4] U. Bandara, M. Hasegawa, M. Inoue, and H. Morikawa. Design and implementation of a bluetooth signal strength based location sensing system. *Radio and Wireless Conference*, Jan 2004.

[5] S. Bryant, A. Forte, and A. Bruckman. Becoming wikipedian: transformation of participation in a collaborative online encyclopedia. *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work.*, Jan 2005.

[6] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College, November 2000.

[7] A. Dey and G. Abowd. Towards a better understanding of context and context-awareness. *CHI 2000 Workshop on the What*, Jan 2000.

[8] Y. Gwon and R. Jain. Error characteristics and calibration-free techniques for wireless LAN-based location estimation. *Proceedings of the 2nd International Workshop on Mobility Management and Wireless Access Protocols.*, Jan 2004.

[9] A. Haeberlen, E. Flannery, A. Ladd, and A. Rudys. Practical robust localization over large-scale 802.11 wireless networks. *The 10th ACM International Conference on Mobile Computing and Networking (MOBICOM).*, Jan 2004.

[10] D. Hahnel, W. Burgard, D. Fox, K. Fishkin, and M. Philipose. Mapping and localization with RFID technology. *Robotics and Automation*, 2004.

[11] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. *Wireless Networks*, Jan 2002.

[12] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, Dec 2001.

[13] J. Hightower and G. Borriello. A survey and taxonomy of location systems for ubiquitous computing. *IEEE Computer*, Jan 2001.

[14] J. Hightower, S. Consolvo, A. LaMarca, and I. Smith. Learning and recognizing the places we go. *Proceedings of the Seventh International Conference on Ubiquitous Computing.*, Jan 2005.

[15] E. Kaasinen. User needs for location-aware mobile services. *Personal and Ubiquitous Computing*, Jan 2003.

[16] T. King, S. Kopf, T. Haenselmann, C. Lubberger, and W. Effelsberg. Compass: A probabilistic indoor positioning system based on 802.11 and digital compasses. In *Proceedings of the First ACM International Workshop on Wireless Network Testbeds, Experimental evaluation and CHaracterization (WiNTECH)*, Los Angeles, CA, USA, September 2006.

[17] M. B. Kjaergaard, G. Treu, and C. Linnhoff-Popien. Zone-based RSS reporting for location fingerprinting. *Pervasive Computing*, 2007.

[18] N. Kuster and Q. Balzano. Energy absorption mechanism by biological bodies in the near fieldof dipole antennas above 300 mhz. *Vehicular Technology*, Jan 1992.

[19] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, J. Tabert, P. Powledge, G. Borriello, and B. Schilit. Place lab: Device positioning using radio beacons in the wild. *Pervasive*, Mar 2005.

[20] N. D. Lane, H. Lu, S. B. Eisenman, and A. T. Campbell. Cooperative techniques supporting sensor-based people-centric inferencing. In *Pervasive*, pages 75–92, 2008.

[21] Z. Li, W. Trappe, Y. Zhang, and B. Nath. Robust statistical methods for securing wireless localization in sensor networks. *Fourth International Symposium on Information Processing in Sensor Networks.*, Apr 2005.

[22] H. Lim, L. Kung, J. Hou, and H. Luo. Zero-configuration, robust indoor localization: Theory and experimentation. *Proceedings of IEEE INFOCOM*, 2006.

[23] K. Lorincz and M. Welsh. Motetrack: a robust, decentralized approach to rf-based location tracking. *Personal and Ubiquitous Computing*, Jan 2007.

[24] C. Morelli, M. Nicoli, V. Rampa, and U. Spagnolini. Hidden markov models for radio localization in mixed los/nlos conditions. *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, 55(4):1525, 2007.

[25] O. Nov. What motivates wikipedians. *ACM Communications*, 50:60–64, 2007.

[26] V. Otsason, A. Varshavsky, A. LaMarca, and E. de Lara. Accurate gsm indoor localization. *The Seventh International Conference on Ubiquitous Computing (UbiComp)*, Jan 2005.

[27] M. Satyanarayanan. Pervasive computing: vision and challenges. *IEEE Personal Communications*, Jan 2001.

[28] R. Want, A. Hopper, V. Falcão, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems (TOIS)*, Jan 1992.

[29] M. Weiser. The computer for the 21 stcentury. *ACM SIGMOBILE Mobile Computing and Communications Review*, Jan 1999.

[30] M. Weiser. Some computer science issues in ubiquitous computing. *ACM SIGMOBILE Mobile Computing and Communications Review*, 3(3), 1999.