Efficient and Accurate WLAN Positioning with Weighted Graphs

René Hansen and Bent Thomsen

Department of Computer Science, Aalborg University
Fredrik Bajers Vej 7E, DK-9220 Aalborg Øst, Denmark
{rhansen,bt}@cs.aau.dk

Abstract. This paper concerns indoor location determination by using existing WLAN infrastructures and WLAN enabled mobile devices. The *location fingerprinting* technique performs localization by first constructing a radio map of signal strengths from nearby access points. The radio map is subsequently searched using a classification algorithm to determine a location estimate. This paper addresses two distinct challenges of location fingerprinting incurred by positioning moving users. Firstly, movement affects the positioning accuracy negatively due to increased signal strength fluctuations. Secondly, tracking moving users requires a low-latency overhead which translates into efficient computations to be done on a mobile device with limited capabilities. We present a technique to simultaneously improve the positioning accuracy and computational efficiency. The technique utilizes a weighted graph model of the indoor environment to improve positioning accuracy and computational efficiency by only considering the subset of locations in the radio map that are feasible to reach from a previously estimated position. The technique is general and can be used on top of any existing location system. Our results indicate that we are able to achieve similar dynamic localization accuracy to static localization. Effectively, we are able to counter the adverse effects of added signal fluctuations caused by movement. However, as some of our experiments testify, any location system is fundamentally constrained by the underlying environment. We give pointers to research which allows such problems to be detected early and thereby avoided before deploying a system.

1 Introduction

Indoor location-based services (LBS) hold promise for a multitude of valuable services to be built and has the potential of substantially leveraging the utility of existing outdoor solutions. Unfortunately, traditional "outdoor" technologies such as GPS and cellular networks do not work reliably and accurately enough in indoor environments. The GPS signal is not sufficiently strong to penetrate most buildings, while the accuracy of cellular positioning is limited by the denseness of GSM antennas in a particular area. A widely researched alternative is to make use of the existing 802.11 (WLAN) infrastructures which nowadays have become ubiquitous for providing wireless communication. Reusing WLAN for

F. Granelli et al. (Eds.): MOBILIGHT 2009, LNICST 13, pp. 372–386, 2009.

indoor positioning is appealing since it avoids the cost of specialized equipment used solely for positioning. Instead, systems can be implemented exclusively in software making indoor positioning a possibility in as large a population as the WLAN infrastructures themselves.

Positioning via WLAN can be done by using signal strength information from available access as the signal strengths vary with spatial changes. However, as radio waves are propagated in an indoor environment, they are also scattered, reflected, and attenuated by the obstacles they encounter. Signals may arrive at the receiver from several paths and when the multiple incoming waves combine at the receiver they can produce a distorted version of the desired waveform, a phenomenom known as *multipath interference* [3]. The end result is a highly unpredictable propagation patterns and the best results in the face of this situation has been achieved with the *location fingerprinting* technique which uses empirically measured signal strengths. The fingerprinting technique works in two phases: First, before a positioning system is deployed, signal strength information is recorded at a number of predefined locations throughout the area to be covered by the positioning system. The measured signal strength values and locations are saved in a database, also called a *radio map*. The process of collecting the signal strength measurements and building the radio map is commonly referred to as the *offline phase*. In the corresponding *online phase*, where the system is operational, a location estimate is obtained by first measuring the signal strengths and then searching the radio map for a closest match to the currently measured signal strength. Different methods can be used to arrive at a closest match and generally these are divided into two main categories: *deterministic methods* and *probabilistic methods*. Deterministic methods build a radio map by saving signal strength information as a scalar value, e.g., the mean signal strength, while probabilistic algorithms store the signal strength distributions from the access points. In the online phase, deterministic methods then estimate a location by comparing measurements by their value while probabilistic methods estimate a location by considering measurements as part of a random process [10,20]. The location fingerprinting technique has produced good positioning accuracy for *static localization*, i.e., location determination of a stationary user. However, users of an LBS service can in general be expected to the moving, for instance users who need navigation to a point of interest, and *dynamic localization*, or tracking a moving user, is more challenging for two reasons. Firstly, movement causes increased fluctuations in the signal strengths. The greater variance means the signal strengths in the radio map are less precise approximations. The result is more dispersed and erratic location estimates and hence the positioning accuracy deteriorates under movements. Secondly, movement necessitates frequent position updates to keep track of the users' current position. Performing localization on a central server and communicating the resulting locations to clients incurs a substantial communication overhead when the update interval is very short. A more scalable solution, which also preserves the privacy of the users, is to perform localization on the users' own devices. Reducing the computational overhead is necessary for constrained devices to meet

the low latency requirements. Moreover, reducing the computational effort saves battery usage and thus helps prolong the lifetime of LBS services.

We present a technique based on a weighted graph model of the indoor environment that tries to improve both the positioning accuracy and computational efficiency of location fingerprinting. The underlying idea is to restrict attention - and hence the search space - to only those locations that are feasible to reach from a previously determined position. The intuition is to improve accuracy by filtering away even nearby locations that are impossible to reach due to obstructions or because they are too far away. Similarly, the search of the radio map is confined to a very small geographic region. We show that this region can be made as small as the positioning accuracy of the underlying algorithm permits and this is provably the smallest region possible.

We apply the technique in two real-world test-beds using a simple deterministic algorithm as the underlying base and only a bare minimum of training. This configuration is chosen in order to demonstrate the robustness of the graph technique. However, the graph technique is general and can be placed on top of any existing location system.

Our results indicate that we are able to achieve similar dynamic localization accuracy to static localization, i.e., the technique is resilient to the adverse effects caused by signal fluctuations. However, the results in one of our test beds also demonstrate that location systems are fundamentally limited by the underlying environment.

The rest of the paper is organized as follows: Section 2 discusses related research in 802.11 based localization with a focus on the research that addresses dynamic localization and computational efficiency. Moreover, the section discusses some of the fundamental constraints of 802.11 based localization. Section 3 introduces the weighted graph technique and gives the implementation details while Section 4 shows the results of the technique in two test beds. Finally, Section 5 concludes the paper.

2 Related Research

The *Radar* project [5], carried out at Microsoft Research, was the first to use commodity 802.11 hardware for indoor localization. Using a deterministic approach with a Nearest Neighbor algorithm this initial research effort was able to achieve a median accuracy of 2.94 meters. The Radar project has since become the defacto standard on which other location systems are measured. Bahl et al. [4] and Smailagic and Kogan [17] extended the Nearest Neighbor approach to produce better results by averaging k nearest neighbors on proximity to produce a location estimate. Recently, focus has shifted to probabilistic methods and several research efforts have achieved accuracies of 2 meters or better with at least 90% probability, i.e., locations estimates are within 2 meters of the actual location 90% of the time [2,11,12,16,18,20,22]. Since probabilistic algorithms draw information from signal strength distributions rather scalar values they can be trained to outperform their deterministic counterpart[19]. However,

environmental factors and tunable parameters such as the building structure, number and placement of access points and number of samples often have a bigger impact than the choice of algorithm as witnessed in several experiments, e.g., [15,12,16,6].

The best-case positioning accuracies have been achieved with static localization when the tracked device remains at the same location. In this situation the signal strengths remain fairly stable and resemble the entries in the radio map. However, movement incurs increased signal strength fluctuation resulting in deteriorating positioning accuracy. Different approaches have been taken to modeling movement. Smagailagic and Kogan [17] propose a time-averaged location convergence technique to try and minimize the jumps of consecutive estimates. Bahl et al. [4] uses a Viterbi-like algorithm for tracking on top of a k-Nearest Neighbor algorithm. Ladd et al. [12] uses a sensor fusion technique together with spatial continuity assumptions to probabilistically reject outliers. The two techniques that are closest to ours are Xiang et al. [18] and Haeberlen et al. [7]. Xiang et al. [18] use a tracking-assistant algorithm implemented as a state machine. Here, each new state (location) is determined by the previous state and the allowed transitions which takes into account the topology of the environment. Haeberlen et al. [7] considers topology and movement in a similar fashion by using a transition graph to describe allowed movement.

Estimating a position by comparing with all entries in the radio map can become a very taxing operation, especially when the operation area increases or when performing localization on a resource-constrained device. Improving efficiency by reducing the search space of location fingerprinting has previously been addressed by using a notion of access point clustering [2,9,20,22]. An access point cluster defines a spatial region where a common set of access points can be heard.

Youssef et al. [22] use an *explicit* clustering technique where a number of clusters are defined by an administrator in the offline phase. Since access points may by intermittently missing, a subset q of the available access is used to define a cluster. This subset is comprised by the access points with the strongest signals as they are most likely to be present. In the online phase the current measurement is sorted in descending order and the q strongest access points are used to determine which cluster to search within to determine a location estimate. The *Locator* system [2] uses explicit clustering for a first-level clustering, and then proceeds with a further k-Means clustering during the online phase based on a previous position estimate and the physical closeness of locations. Similarly, the *Ariadne* system [9] uses k-Means clustering, but instead clusters a set of candidate positions with smaller mean square errors. The largest cluster is chosen and its center is picked as the location estimate.

Rather than explicitly clustering locations in the offline phase, the *Horus* system [20] uses an *implicit clustering* technique to defer clustering until the online phase. In the online phase the current measurement is once again sorted in descending order. Then, the algorithm proceeds incrementally: For the strongest access point, the probability of each location containing that access point is

calculated. If the probability of the most probable location is significantly higher than the probability of the second most probable location according to a threshold parameter, this location is returned as the location estimate. Otherwise, the next access point in the sorted access point list is consulted. The process continues until a position estimate can be delivered.

A comparison of the two clustering approaches is performed by Youssef and Agrawala [21] which found that explicit clustering results in better positioning accuracy than implicit clustering but incurs a larger computational overhead.

3 Using Weighted Graphs for Efficient and Accurate Tracking

Weighted graphs are a fundamental data structure in computer science and have been applied in a host of different domains, including GIS mapping and road-network modeling. Similarly, weighted graphs are a natural formalism for modeling the topology of an indoor environment. As we will demonstrate weighted graphs can be further used to help provide accurate and efficient dynamic localization. We build a weighted graph model of the indoor environment by modeling fingerprinted locations as nodes with weighted edges to capture the distance between physically adjacent locations. Such a graph model can easily be built on top of an existing system or as part of the offline phase of a new location system.

In the online phase the weighted graph is utilized to provide accuracy and efficiency in the following way: Given a previously estimated position, the attention (and search space) is confined to *feasible* locations. These are the locations that are reachable within the update interval of consecutive location estimates by a user traveling at an assumed maximum speed. Thus, feasible locations can be found by performing a breadth-first search within the range ($max\_speed\_per\_second \times update\_interval$). With dynamic localization this range is not very large. Assuming an update interval of one second and a maximum pedestrian speed of three meters per second, the reachable range is 3 meters. In practice, this would in many cases correspond to only the neighboring nodes given the difficulty of distinguishing signal strengths at a finer granularity. As a result, the search space is very small and intuitively accuracy is improved by filtering away even nearby locations that cannot be reached.

Due to the noisy nature of the wireless channel this scheme is, of course, somewhat idealized and there is an inherent risk that a localization system could get "stuck" at a location. This would happen if the actual and estimated locations are not connected by a path, e.g., if they are separated by a wall with no doors connecting them. In order to handle this we introduce so-called *radius-nodes*. Radius-nodes are defined as the set of all non-connected nodes located within a specified euclidian distance of a node. Radius-nodes are used to account for estimation error to non-connected nodes of the actual location. As such the range of radius-nodes is determined by the precision of the underlying positioning system. As an example, if the system can deliver position estimates with 100% certainty within 10 meters, correspondingly the radius should be set to 10 meters in order

to consider all relevant nodes. It follows that this is also the provably smallest geographical region of the radio map that needs to be searched since a smaller region would fail to consider all relevant nodes. Lemelson et al. give algorithms to estimate 802.11 positioning error which can be used to ascertain the range of radius-nodes[13]. Radius-nodes are implemented by augmenting each node with a list of its radius-nodes. Radius-nodes are found using a range query centered around the node in question. The range of radius-nodes can be set to different sizes in different parts of the building to reflect varying localization error.

In the online phase radius-nodes are used as a fall-back mechanism when the idealized scheme fails. The governing rule is to prefer connected nodes to a previously estimated position as this provides smooth tracking and avoids erratic jumps through obstructions. Only if there are strong indications that the currently selected track is wrong are these logical jumps allowed. To achieve this we operate with two distinct search spaces: A *primary search space* which consist of the reachable locations from a previously estimated location l and a *secondary search space* consisting of the radius-nodes of l. Both search spaces are searched and at any point a location estimate from the primary search space is preferred. However, if strong evidence suggest that a non-connected node is indeed more likely a "shift" can be made. A shift entails selecting the most probable non-connected node as the location estimate and adjusting the search space such that the primary search space now becomes centered around the new node. The notion of making a shift is illustrated in Figure 1.

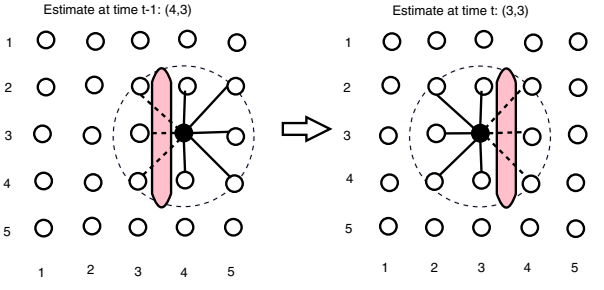


Fig. 1. A position estimate at time *t-1* (filled circle) has a primary search space (nodes connected by solid lines) and a secondary search space (nodes connected by dashed lines). At time t a radius-node is selected as the new position estimate. A "shift" is made by updating the primary and secondary search. The dashed circle indicated the range of radius nodes.

Evidence that a shift needs to be performed can come from a history of measurements. For example, if two or three consecutive estimates has been to radius-nodes this could indicate that a shift is needed. The range of radius-nodes, however, should be extended in accordance with the length of the history. So if a history-length of three seconds is used, the radius should be extended by $3 \times max\_speed\_per\_second$ since the user can move this distance further away

from the correct location before a shift is made. We advice using a history-length of two seconds, and making a shift if two consecutive best estimates has been to radius nodes. By using a short history-length the search space is kept small and at the same time the system can quickly react to seemingly false assumptions. Conversely, a longer history-length may yield even stronger indications but has a correspondingly longer reaction time.

3.1 Algorithm and Data Structures

We now proceed to giving the details of the method that implements the graph technique. The method has been implemented in C# and is presented in a combined C#/pseudo-code syntax in Listing 1 to improve readability. The method is a template method where the individual steps can be customized to fit an existing system and application domain by adjusting the underlying parameters.

```
1  public void estimatePosition() {
2    Measurement meas;
3    Estimate est;
4    Estimate radiusEst;
5    List<Node> primSpace = graph.Nodes;
6    List<Node> secSpace;
7    History hist = new History();
8
9    while (ONLINE_RUNNING)
10   {
11     meas = MeasureSS(UPDATE_INTERVAL);
12     est = Compare(meas, primSpace);
13     nodes = FeasiblePositions(est);
14     secSpace = est.RadiusNodes;
15     radiusEst = Compare(meas, secSpace);
16     history.add(est, radiusEst);
17     if (history.correctionNeeded())
18       nodes = feasiblePositions(radiusEst);
19   }
20 }
```

Listing 1. Position estimation with radius-nodes

The *estimatePosition()* procedure is called when location estimation commences. We start by describing the data structures involved. The *Measurement* class encapsulates a signal strength measurement and the variable `meas` holds the current measurement. The `Estimate` class represents a position estimate - a node - along with the node's *score* which is a measure of how good the estimate is, e.g., highest probability or shortest distance. *Node* objects correspond to nodes in the graph and the objects `est` and `radiusEst` represent the currently

best scoring node in the primary and secondary search space, respectively. The primary search space is represented in the `primSpace` list while the secondary search space is kept in the `secSpace` list. Finally, the `History` class represents a history of measurements (a queue of recent measurement), and a criteria for performing a correction. Performing a correction corresponds to making a shift as described in Section 3. The `History` allows for different queue lengths and criteria for correction to be set. It can be noted that in the code above the primary search initially consists of all nodes in the graph (line 5). In a large deployment, however, we would instead use one of the access point clustering techniques as described in Section 2 to reduce the initial search space.

The loop (lines 9-19) defines what happens during location determination: The signal strength is measured and saved into the `meas` object (line 11)[1]. The measurement is compared with the nodes in the primary search space (`primSpace`) using an underlying classification algorithm which results in in a position estimate (line 12). The primary search space is updated to be the set of reachable locations from the estimated node (line 13) and the secondary search space is updated to be the radius nodes of the estimated node (line 14). The signal strength is then compared with the radius nodes resulting in yet another estimate - to the best-scoring radius-node (line 15). Both estimates are added to the history (line 16) and if the history indicates that a correction is needed, a shift is made (line 17-18).

4 Experiments

The graph technique has been implemented in a prototypical LBS application running on the Windows mobile platform. This section discusses the setup and results from two separate experiments that were conducted with the application in order to test the robustness of the technique in different settings.

The experiments were performed using an HP Ipaq H5550 Pocket PC with an Intel XScale 400 Mhz processor. Experiments were conducted using both the in-built wireless Network Interface Card (NIC) as well as an SDIO NIC from Socket. An API from OpennetCF.org was used to query the NICs for the detected signal strengths.

In both experiments, the radio map was built by saving the average signal strength value of only ten measurements at each location. In the online phase a single measurement was used and the update interval was set to one second.

The system made use of a simple deterministic Nearest Neighbor algorithm similar to the one used in the original Radar project The choice of a bare minimum of signal strength samples coupled with a simple deterministic algorithm was made in order to "stress test" the robustness of the graph technique. A more advanced probabilistic algorithm coupled with more samples might have improved the accuracy further but this would also mean that the concept of radius-nodes would not be tested to its limits.

[1] OpennetCF provides an API for the .NET Compact Framework for querying signal strength information [1].

The experiments were performed by pressing a start/stop button which starts and stops the loop described in Listing 1. The results were recorded by logging all estimates produced during the test. This approach was taken, in contrast to clicking the actual position while the test was underway, as it gave a more realistic view of how the technique handles movement. Instead of stopping briefly to mark the current location, the test could be performed with a continuous walk. The assumed maximum speed were in both experiments set to three meters per second.

4.1 Aalborg University

The first test was performed in a wing of Aalborg University. A total of 16 access points could be heard throughout the building and at any one location the number of access points ranged from four to eight. The weighted graph was built by placing a node in offices and at locations where there were "forks" in the environment (the experiments were conducted at a holiday period which accounts for only some of the offices being included). The nodes were approximately three meters apart corresponding to the size of an office. The model consisted of 71 nodes and 75 edges and can be seen in Figure 2.

The figure also shows the radius that was chosen based on the observed static localization accuracy and chosen history-length. The worst-case static ac-

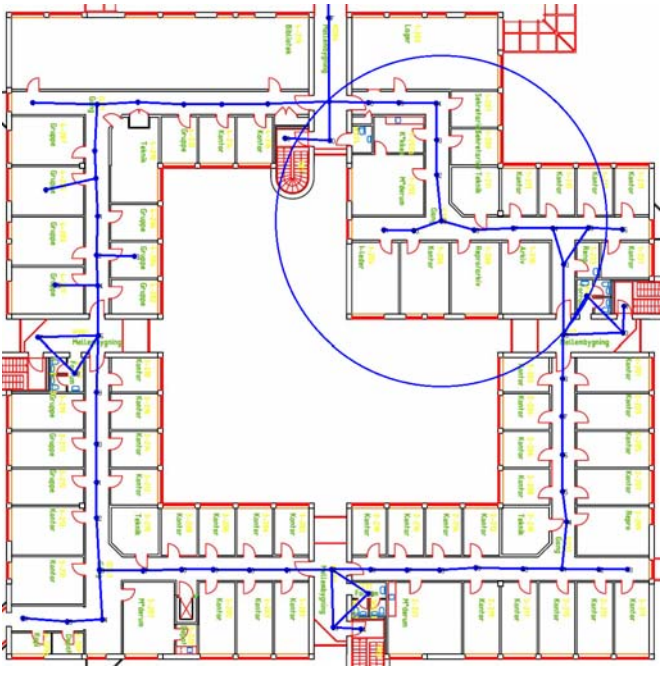


Fig. 2. Wing at Aalborg University modeled as a weighted graph. The size of the radius-nodes can be seen in the top-right circle.

curacy was nine meters and two consecutive best-scoring radius-estimates was used as the correction criteria which led to an additional three meters being added to the radius.

We completed eight walks around the building with both the graph technique and the original Nearest Neighbor algorithm. We used four start/stop points in the upper, lower, left, and right side of the building and tested by walking in both directions. The graph- and original technique were tested back to back. Immediately after completing a walk with the graph technique we took a tour with the original technique. This was done to avoid temporal differences in the comparison. The results of the tests can be seen in Figure 3.

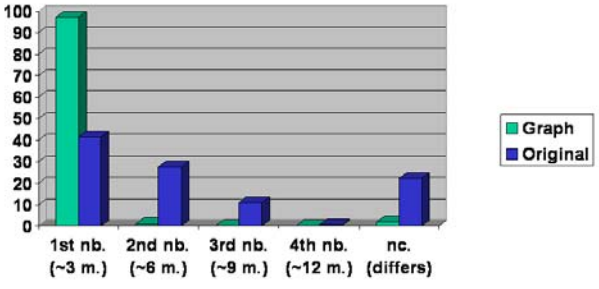


Fig. 3. Dynamic localization accuracy in Aalborg University

The first column represents location estimates that were at, or at most one node (approximately three meters) away from the actual location. Columns two through four represent location estimates that were two to four nodes away from the actual location, with the corresponding accuracy shown in parentheses, while the last column indicates those results that were at a non-connected node to the actual location. As can be seen the weighted graph technique massively outperforms the original Nearest Neighbor algorithm whose performance degrades significantly when moving. After the head-to-head comparisons a number of tests were run with the graph technique alone to test the robustness further. In these tests we played the role of an indecisive user who changes direction several times. This did not affect the accuracy adversely nor did using the either of the NICs. Since the activity level in the building was somewhat idle at the time of the experiments it cannot be ruled out that the accuracy would have been lower had there been more people and offices open. However, in the tests the three offices that were open were not selected as location estimates in the walks even though the doors were open. Instead, the few times the graph technique made a correction was when the operator passed by the midsections which can be found in the middle left, right, top, and bottom of the building. In most cases a wrong estimate here was resolved by the system "catching up" but sometimes the system chose to cut the corner.

Table 1. Computation time in Aalborg University

Card / Method	Graph	Radar
In-built	10-20 ms.	ca. 35 ms.
SDIO	30-40 ms.	ca. 60 ms.

The computational performance of the two techniques is summarized in Table 1.

Using the device's in-built NIC the computation time with the graph technique ranged from 10 to 20 milliseconds while the original Radar algorithm produced more stable results at around 35 milliseconds. The greater variance with the graph technique is due to a varying number of radius-nodes in different parts of the building. Using the Socket SDIO NIC added another 25 milliseconds to the computation time which is attributed to I/O operations. In comparison, we can see that the graph technique is approximately twice as fast as the Nearest Neighbor algorithm which searches the entire radio map. The graph technique searches less than half of the radio map, which can be seen by observing the radius in Figure 2, but adds a bit of extra programming logic which is the reason why the difference is not greater. However, since the computational complexity of the graph technique is constant the difference will increase further in even larger areas.

4.2 Randers Public Library

Experiments were also conducted in Randers Public Library. This section does not include a formal treatise of the results as the environment posed a pathological case for indoor positioning. Rather, we discuss the experiences from the experiments. The library had four access points installed but in certain parts of the building sometimes only two of the access points could be registered. A weighted graph model was created resulting in 142 nodes and 214 edges which can be seen in Figure 4.

As can be seen the weighted graph does not extend to the far right. This is because the building map was slightly outdated. As a result this section of the building was left out of the experiments. Upon building the radio map samples were taken at nine different locations in order to get a picture of the static localization accuracy. This revealed problems pertaining to delivering accurate positioning. In the left side of the building - at 60 out of the plotted 142 nodes it turned out - estimates jumped very far, as far as from one end of the building to the other. In this section it was observed that frequently only two of the access points could be registered so the problem was initially tried amended by using several samples in an effort to hear more access points. By using several samples we managed to pick up all four access points. However, this did not improve the accuracy. An analysis of the data in the radio map showed the signal strengths of three of the access points in this region to be highly overlapping. As Figure 5

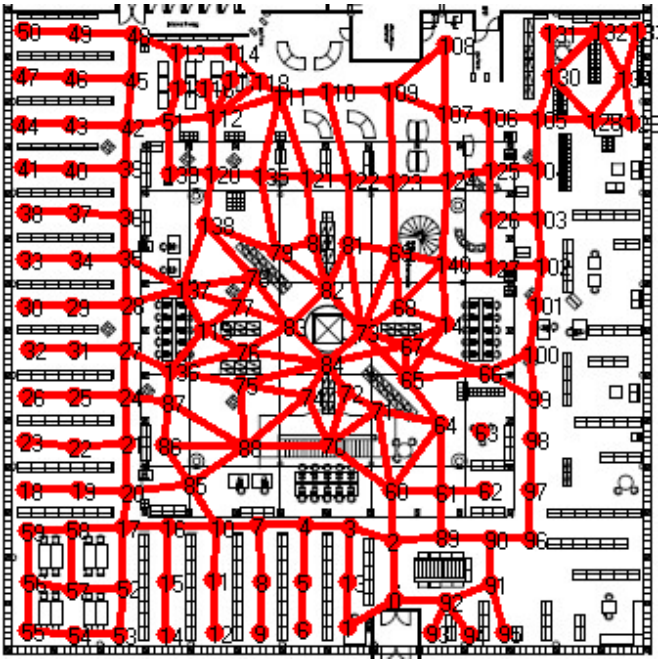


Fig. 4. Randers Public Library modeled as a weighted graph

shows especially two of the access points were affected showing a variance within only 10 dBm. Since it was not uncommon to see variances of 10 dBm simply by changing direction this made it impossible to perform sensible localization in this part of the building. In the middle- and right side of the building, however, the static localization accuracy was acceptable. The accuracy was lower than in Aalborg - more estimates were farther away from the actual location but the worst-case accuracy remained the same. As a result, the same setup as before was used.

The difference in dynamic localization accuracy was even more notable than in Aalborg University and the effect of movement was substantial as witnessed in Figure 6.

As can be seen, the graph technique initially does a very good job of confining the location but in the lower left corner it gets stuck whereafter it starts jumping back and forth while the user continues to the upper left corner. This illustrates the accuracy in a nutshell: While refraining from the left side of the building accuracy was substantially improved but any attempts to use the graph technique in the left side resulted in the system getting stuck. This was resolved only by changing the radius to the 40 meters locations could jump in the left side! Outside the left region the graph technique distinguished locations on opposite sides of bookshelves very well. In the mid-region where the obstructions were formed by low bookshelves (<1.5 meters high) and tables the system some times chose

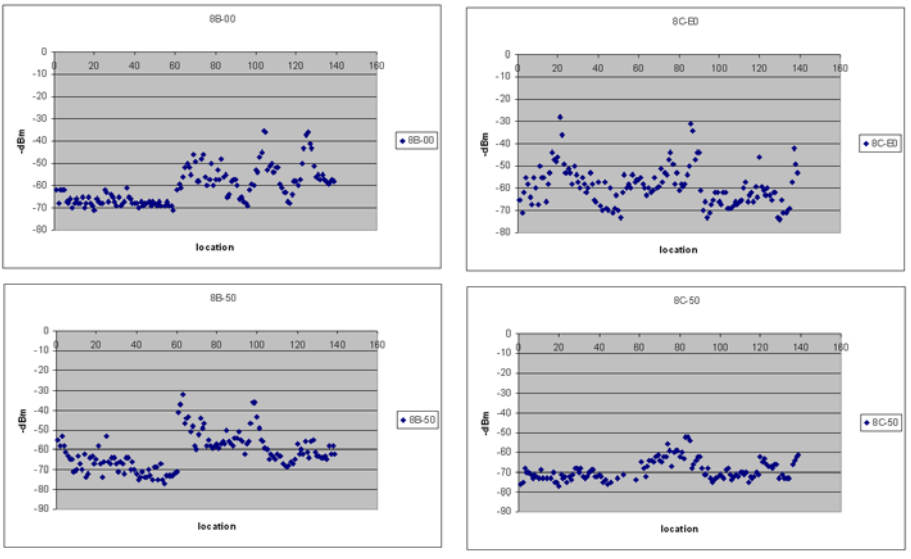
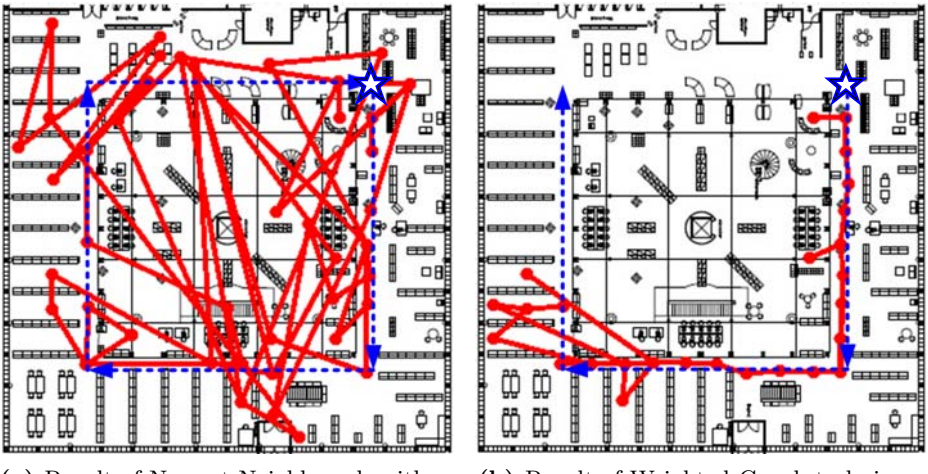


Fig. 5. Signal strengths from the four access points at Randers Public Library



(a) Result of Nearest Neighbor algorithm **(b)** Result of Weighted Graph technique

Fig. 6. Comparison of a walk in Randers Public Library. The star symbol indicates the starting point and the dashed line the actual route walked. The solid lines connect consecutive location estimates.

the wrong path around an obstruction but stayed within close proximity of the actual location in contrast to the Nearest Neighbor algorithm which exhibited very large estimation jumps when moving. The experiments in Randers Public Library are a testament that 802.11-based location systems are fundamentally

limited by the available infrastructure. In some cases performance can only be improved to an acceptable level by restructuring or adding new access points. Although we were not free to modify the wireless infrastructure in this experiment, Lemelson et al. [13] presents a strategy for predicting the estimated location error based on the signal strength patterns in the radio map. The strategy clusters together locations with similar signal properties and the estimated error of a region is derived from the distance of the cluster of that region. This strategy allows potential trouble areas to be identified before deployment and appropriate countermeasures to be taken.

5 Conclusion

In this paper we presented a weighted graph technique for improving the computational efficiency and accuracy of dynamic localization (tracking of moving users) by using available 802.11 equipment. The weighted graph technique was used to impose topological constraints on the location system and improve the accuracy and efficiency of an underlying classification algorithm by only considering feasible location candidates based on the users' previous location and an assumed maximum traveling speed. The technique was found to provide a substantial improvement in dynamical localization accuracy by disregarding location estimation outliers caused by movement. At the same time the search space was confined to a very small geographic region which facilitates building scalable systems where the computation is performed on the users' own mobile devices. However, as experiments in one of the tests demonstrated the usability of an LBS service is fundamentally limited by the underlying wireless infrastructure that supports it. We provided pointers to research which allows such ailments to be discovered early and to be avoided before deploying an indoor location system.

References

1. Opennetcf.org, `http://www.opennetcf.org/home.ocf`
2. Agiwal, A., Khandpur, P., Saran, H.: Locator: location estimation system for wireless lans. In: WMASH 2004: Proceedings of the 2nd ACM international workshop on Wireless mobile applications and services on WLAN hotspots (2004)
3. Alexander, B.: 802.11 Wireless Network Site Surveying and Installation. CISCO Press (November 2004)
4. Bahl, P., Balachandran, A., Padmanabhan, V.: Enhancements to the RADAR User Location and Tracking System. Microsoft Research Technical Report (February 2000)
5. Bahl, P., Padmanabhan, V.N.: Radar an in-building RF-based user location and tracking system. In: INFOCOM 2000, Tel Aviv, Israel, pp. 775–784 (2000)
6. Elnahrawy, E., Li, X., Martin, R.P.: The limits of localization using signal strength: a comparative study. In: Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004, pp. 406–414 (October 2004)

7. Haeberlen, A., Flannery, E., Ladd, A.M., Rudys, A., Wallach, D.S., Kavraki, L.E.: Practical robust localization over large-scale 802.11 wireless networks
8. Hightower, J., Borriello, G.: Location sensing techniques. Technical report. companion to "location systems for ubiquitous computing" appearing on pp. 57–66, august issue 2001 of ieee computer magazine, University of Washington, Computer Science and Engineering, Box 352350, Seattle, WA 98195 (July 2001)
9. Ji, Y., Biaz, S., Pandey, S., Agrawal, P.: Ariadne: A dynamic indoor signal map construction and localization system. In: MobiSys 2006 (2006)
10. Kjærgaard, M.B.: A taxonomy for radio location fingerprinting. In: Hightower, J., Schiele, B., Strang, T. (eds.) LoCA 2007. LNCS, vol. 4718, pp. 139–156. Springer, Heidelberg (2007)
11. Krumm, J., Horvitz, E.: Locadio: Inferring motion and location from wi-fi signal strengths. Mobiquitous, 4–13 (2004)
12. Ladd, A.M., Bekris, K.E., Rudys, A., Marceau, G., Kavraki, L.E., Wallach, D.S.: Robotics-based location sensing using wireless ethernet. In: MOBICOM 2002, September 23-26 (2002)
13. Lemelson, H., Kjaergaard, M.B., Hansen, R., King, T.: Error estimation for indoor 802.11 location fingerprinting
14. Battiti, M.B.R., Villani, A.: Statistical learning theory for location fingerprinting in wireless lans. Dit-02-086, University of Trento, Informatica e Telecomunicazioni (October 2002)
15. Roos, T., Myllymäki, P., Tirri, H., Misikangas, P., Sievänen, J.: A probabilistic approach to wlan user location estimation. International Journal of Wireless Information Networks 9(3) (July 2002)
16. Saha, S., Chaudhuri, K., Sanghi, D., Bhagwat, P.: Location determination of a mobile device using ieee 802.11b access point signals. In: IEEE Wireless Communications and Networking Conference (WCNC 2003), March 2003, pp. 1987–1992 (2003)
17. Smailagic, A., Kogan, D.: Location sensing and privacy in a context aware computing environment. IEEE Wireless Communication 9(5), 10–17 (2002)
18. Xiang, Z., Song, S., Chen, J., Wang, H., Huang, J., Gao, X.: A wireless lan-based indoor positioning technology. IBM J. Res. Dev. 48(5/6), 617–626 (2004)
19. Youssef, M., Agrawala, A.: On the optimality of wlan location determination systems. Technical Report UMIACS-TR 2003-29 and CS-TR 4459, University of Maryland, College Park, 2003 (2003)
20. Youssef, M., Agrawala, A.: The horus wlan location determination system. In: MobiSys 2005: Proceedings of the 3rd international conference on Mobile systems, applications, and services, pp. 205–218. ACM Press, New York (2005)
21. Youssef, M., Agrawala, A.: Location-Clustering Techniques for WLAN Location Determination Systems (2006)
22. Youssef, M.A., Agrawala, A., Shankar, A.U.: Wlan location determination via clustering and probability distributions. In: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom 2003) (2003)