Cooperative location-sensing for wireless networks

Charalampos Fretzagias Maria Papadopouli[*]

Department of Computer Science, University of North Carolina at Chapel Hill

Abstract

We present a robust location-sensing system that enables devices to estimate their position in a self-organizing manner without the need for an extensive infrastructure or training. The algorithm uses a grid representation that allows the system to easily incorporate external information (such as GPS, application-dependent semantics, geographic and signal strength maps) to improve the accuracy of the position estimation. Depending on the capabilities of the devices, the system can operate in a distributed, centralized, or hybrid fashion. In the case of devices with limited capabilities, a centralized or hybrid approach can be computationally inexpensive and therefore more appropriate. Also, hosts may cooperate and share positioning information. By allowing the cooperation among hosts, the system can improve its accuracy and provide more flexibility. We designed and implemented the system and analyzed its performance via simulations and actual measurements. For example, we used it in an indoors area of 45x38 m^2 with two IEEE 802.11 access points (APs) in the Department of Computer Science. By using a small training set the algorithm can predict the location of 50% of hosts with an error of at most 0.9 m and 85% of the hosts with at most 2.5 m accuracy. Furthermore, we investigated the impact of the density of landmarks, degree of connectivity, range, error in the distance measurements, and grid resolution on the accuracy of the location sensing via simulations. With an error of 5% of the transmission range, 20% of the hosts landmarks, and average network connectivity above 7, the average error is less than 2% of the transmission range.

1 Introduction

The effect of Moore's law is transforming a niche technology into a ubiquitous one, expanding the innovations in an increasingly networked world. There is growing interest in the transportation industry to equip vehicles with navigation tools and location-based services; in the medical community with patient monitoring and assistive technology; in the entertainment industry and in emergency situations for disaster relief. To support location-dependent services, a mobile device needs to estimate its position. For example, the Hertz NeverLost GPS-enabled system allows the user to compute a route that will then guide him/her. However, GPS does not work everywhere. In particular, the technology typically breaks down near obstacles such as trees and buildings and does not work indoors. Even ground-based cellular systems' radio signals can be occluded, or they might bounce off nearby objects, making localization difficult or impossible.

Location-sensing systems can be classified according to their dependency and use of an infrastructure, signal modalities, and methodology for estimating distance [7]. Several location sensing systems use radio, infrared, sonar, or vision to infer the position of a mobile device. Recently IEEE 802.11 networks have become widely available

[*]e-mail: fretzagi,maria@cs.unc.edu

in universities and corporations providing wireless Internet access. More and more such networks are becoming available in airports, hospitals, shopping centers, and other public areas [16, 2]. Its low cost and the complexity advantages of using it for both communication and positioning makes IEEE 802.11 an attractive choice. There are location sensing systems [3, 9] that use an infrastructure of IEEE 802.11 APs. They apply probabilistic inference of position mechanism using the RF emission from APs measured by the IEEE 802.11 cards to track the device position. They build a map of the environment by sampling the space and gathering data at various predefined checkpoints of the indoor environment. Other works [14] combine different signal modalities, such as ultrasound from deployed sensors and radio from the IEEE 802.11 infrastructure for location estimation.

There are some recent research efforts on location-sensing systems that operate in an ad hoc network [15, 12, 8, 4]. These systems assume an ad hoc network and cooperation among hosts in the network. Hosts cooperate and exchange positioning information to estimate their location. Like these approaches, our system also enables devices in a self-organizing manner to cooperate with each other. Hosts gather positioning information periodically from other cooperative devices and estimate their distance from their neighboring peers. They can refine their measurements iteratively as they incorporate new information from their neighbors. The main differences of our work from [15, 12] is the grid-based representation of the environment and positioning estimation algorithm. More specifically, the grid representation allows the system to easily incorporate external information (such as GPS, application-dependent semantics, floor plan, signal strength maps) to improve the location estimation. In addition, depending on the characteristics of the environment, it can operate in a distributed, centralized, or hybrid fashion.

In many situations due to environmental, cost, and regulatory barriers the deployment of a wireless infrastructure (for communication or location sensing) is not feasible. In addition, the dynamic characteristics of the environment and the nature of the signal propagation impose important challenges for the design of a scalable, easily deployable, and computationally inexpensive location-sensing system. Our goal is the design a location-sensing mechanism that is

1. robust enough to tolerate multiple network failures (such as device failures or disconnections) and changes in the environment due to host mobility,
2. easily extensible to incorporate application-dependent semantics or external (location-related measurements) information,
3. computationally inexpensive so that devices with limited capabilities (such as PDAs or sensors) can participate in the network,
4. suitable for indoor and outdoor environments,
5. scalable, inexpensive, and easily deployable without the need for extensive training and specialized infrastructure.

We build a system that can operate in a very diverse range of applications. It is easy and cost effective to deploy and maintain, yet it remains competitive in its accuracy compared to the best published results on location-sensing in mobile computing. Section 2 presents the location-sensing algorithm and the different architecture designs. Section 3 discusses the performance results via simulations and actual measurements. Section 4 discusses related work on location-sensing using an infrastructure or in an ad hoc network. Finally, in Section 5 we discuss our main conclusions and future work plans.

2 Voting algorithm

The objective of the proposed system is to enable devices to determine their location in a self-organizing manner and without the need of an extensive infrastructure or training. It can be used with a broad range of location-dependent applications and devices with different computing capabilities. For a broad range of location-dependent applications, the characteristics described in Section 1 are desirable and shape our vision for this system. In the following paragraphs we describe the main location-sensing algorithm and the communication among hosts. Then, we discuss several adaptations of this algorithm for environments with different characteristics or requirements (such as sensor networks with stationary constrained devices, environments with an infrastructure of APs and mobile wireless-enabled laptops and PDAs, cars in a highway enabled with location-dependent applications).

Some devices in the network may have apriori knowledge of their location. We will refer to them as *landmarks*. Landmarks can be stationary devices configured with their position or can be devices capable of estimating their position via another location-sensing mechanism such as GPS. Instead of attempting to solve this system analytically, we use a *grid-based* representation of the terrain. Each cell in the grid is associated with a value that indicates the likelihood that the host is in that cell. We denote as G_k the grid of the host k and as $v(i,j)$ the probability that the cell $(i,j) \in G_k$ is the position of host k. A host tries to position itself (on its local grid). We use the signal strength of wireless transmissions to estimate the distance $d_{m,n}$ between two hosts m and n with grid coordinates (x_m, y_m) and (x_n, y_n), respectively. We define $G_{m,n}$ as the set of cells equal to $\{\ (x,y) \mid \ (x,y) \in G_m \quad d_{m,n} - \epsilon_l \leq \sqrt{(x_m - x)^2 + (y_m - y)^2} \leq d_{m,n} + \epsilon_u\}$. We will refer to $G_{m,n}$ as the *distance estimation* of the two hosts m and n in the grid G_m, where ϵ_l and ϵ_u indicate the accuracy of the estimation and in general depend on distance $d_{m,n}$. The precision of the distance estimations depends heavily on the wireless networking technology used. We will discuss this issue further in Section 3.

The *positioning information* of a node includes its id, maximum wireless range, position (if known or computed). We refer to this broadcast update message as *positioning message*. Hosts disseminate their positioning information to the network. The communication protocol depends on the architecture approach that is used (centralized, distributed, and hybrid). In order to determine its location, each host h will have to gather distance estimations and then attempt to compute its own location using the following algorithm:

1. Initialize the values of the grid G_h with all cells containing zeros.
2. For each distance estimation to a host k with known location, perform the following steps:
3. (a) Transform the coordinates of host k and numbers $d_{k,h} - \epsilon_l$ and $d_{k,h} + \epsilon_u$ to the coordinate system of the grid.
   (b) Determine the region of the grid, i.e., set of cells that satisfy the distance estimation of h and k, $G_{h,k}$.
   (c) Increase the value of the cells in $G_{h,k}$ by v_k.
4. Assess the values of the cells in the grid and accept or reject the attempt for location sensing.

This algorithm is essentially a “voting” process. Host h casts ”votes” into the cells of the grid on behalf of other hosts with known location. Votes have different *weights* based on the host from which the distance is estimated. For example, host k depending on the confidence of its positioning estimation, has a voting weight v_k. At the end of the second step of the above algorithm, the more voting weight (value) a cell has acquired, the more hosts think that it is a likely location for host h. The set of cells in the grid with a maximal value defines a possible region where the host h is located. If this set of cells satisfies certain conditions, the centroid of the area defined by these cells

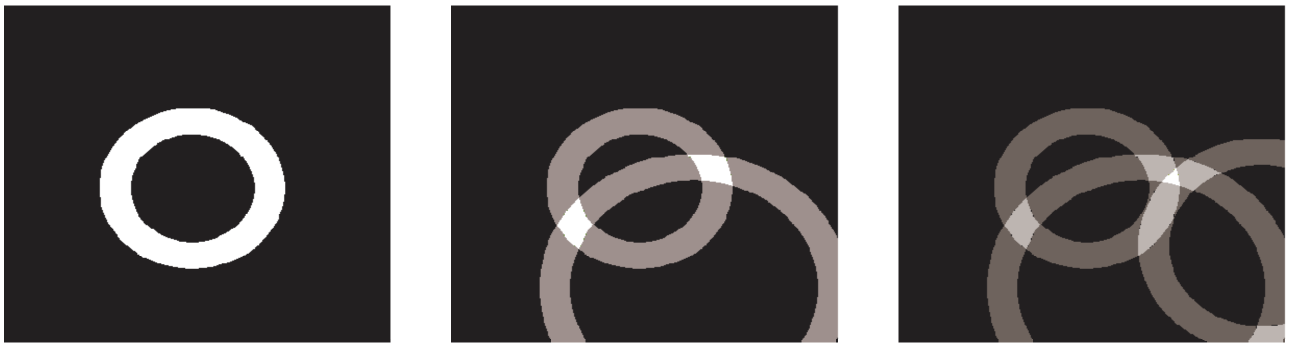


Figure 1: Accumulation of votes on the grid cells of a host at different time steps.

corresponds to the estimated position. Figure 1 shows a snapshot of the grid as three landmarks vote on the location of an unsolved host. The brighter an area, the more voting weight has been accumulated on the corresponding grid cells. The brightest area corresponds to a potential solution.

Landmarks or nodes that computed their own location first determine to some extent the location of other hosts. Therefore, their positioning errors will be propagated in the network, affecting the positioning estimations of hosts that use their votes. To minimize the impact of these errors, a potential solution has to satisfy the following two conditions in order to be accepted as a valid location:

- The number of votes in each cell of the potential solution must be above a threshold. We refer to this threshold as the Solution Threshold (ST).
- The size of the grid region (i.e., number of cells with maximal value) that contains the potential solution must be below a threshold denoted as the Local Error Control Threshold (LECT).

In effect, ST controls how many hosts with known location must “agree” with the proposed solution. Setting a high ST value reduces the error propagation throughout the network, but delays the positioning estimation. On the other hand, LECT determines the precision of each step of the proposed location sensing algorithm. Note that the diameter of the grid region (i.e., maximum Euclidean distance of cells with a maximal value) can be another metric for filtering the local error. Once a potential solution has been accepted, the algorithm computes the centroid of the grid area that corresponds to this solution. This centroid is the estimated location of the newly solved host. By selecting the centroid, we minimize the maximum possible error (given that the correct solution lies somewhere in that grid). Nevertheless, by narrowing down an area to one point, the algorithm potentially introduces an error. If the area of the potential solution is above the LECT, its corresponding centroid will potentially introduce an unacceptable error. In that case, the potential solution will be rejected. The fundamental idea behind both thresholds is that it is likely that other hosts in the network will have higher precision in their position estimates. Therefore, more distance estimations of hosts with known location will be available to increase the voting weight or narrow down the area of the potential solution within acceptable standards. The system can adaptively relax both thresholds after a number of rejected solutions. The values for both the ST and LECT depend on network characteristics such as the density of hosts, the ratio of landmarks and hosts with unknown location and the accuracy of the distance estimations. Next we present the centralized, distributed, and hybrid approaches for the deployment of the voting algorithm.

Centralized architecture

The centralized approach is very easy to implement, has minimal communication overhead, and does not require the nodes to perform any computations. However, it requires the infrastructure of a computational server that will solve the locations of hosts on their behalf. Thus, this approach can only scale as much as the available computational power of that central server for a given region. The centralized location sensing is composed of the following steps:

1. Each node broadcasts a positioning message when it joins the network. This step also takes place periodically to ensure that mobile or new hosts can join the system smoothly. The period is a parameter that depends on the frequency at which the central server can afford to solve and update the locations of the network nodes, frequency at which nodes join the network, and traveling speed of mobile hosts.
2. Hosts listen for broadcasts from other nodes. For each received positioning message, a host creates a distance estimation from that host based on perceived signal strength. Then, it forwards its distance estimations to the centralized server.
3. The server runs the voting algorithm as described earlier on behalf of every host from which it received distance estimations. Then, it informs each host of its estimated location.

Distributed architecture

Although the centralized approach has several advantages, it has two significant drawbacks. It requires an infrastructure and does not scale as the number of nodes in the network or the size of the terrain/grid increases. For applications where nodes do not have strict power conservation restrictions and are capable of carrying out more complex computations, a distributed algorithm can be used. In the distributed approach, each host maintains a database that contains the id, maximum transmission range, estimated location and confidence (if available), and distance estimation for all hosts it has received broadcasts from. This database is broadcasted whenever it is updated and periodically to make sure new nodes can be added to the network while the algorithm is running. Each host tries to solve only its own location using the voting algorithm as described earlier. If successful, it updates its database with its own location and continues to broadcast it periodically. These broadcasts are single-hop broadcasts and not flooding. In the centralized architecture the communication overhead of each node is the total number of forwarding messages to the server, $O(d)$, where d is the diameter of the network. On the other hand, the distributed approach requires hosts to broadcast their databases periodically and whenever there is a database update.

Hybrid architecture

In the hybrid architecture, some nodes estimate their own location using the distributed approach, while others "elect" a server, and use the centralized approach. Hosts using the centralized approach need to perform the following additional tasks:

1. Broadcast their position when they receive it from the server. In that way, hosts using the distributed architecture or that are beyond the transmission range of the centralized server can get this information.
2. Each node forwards information received from hosts that operate using the distributed architecture to its centralized server.

A smart space with sensors, laptops, and PDAs can use the hybrid approach for its location sensing. PDAs and sensors can use a server to estimate their location (i.e. centralized approach), while laptops can compute their own location in a distributed manner. Furthermore, laptops can volunteer to act as server hosts for their neighboring PDAs, thus eliminating the need of any infrastructure.

Incorporating external information in the voting algorithm

As mentioned earlier, one useful characteristics of the proposed algorithm is that it is not application-dependent. However, the system can incorporate application-dependent or external information in the algorithm to further refine the accuracy of the measurements. This can take place in the form of virtual landmarks casting additional votes of their own. In our example of the voting process in Figure 1, we used a disk-shaped model for the wireless transmission range. However, the shape of the regions being accumulated on the grid through voting does not need to be disk-shaped. Actually, in an indoor environment, the placement of obstructions usually causes a degradation is the signal strength. Hence, a more irregularly shaped signal propagation model is more realistic.

The algorithm can also easily incorporate map information to filter out regions where a device cannot be located. The map information can be spatial or dependent on application semantics. For example, in the case of a floor plan, areas within walls are not likely positions for nodes. In the case of devices equipped with some external location-sensing mechanisms (such as GPS, accelerometer, acoustic sensors), the proposed system can acquire their positioning information and precision to further refine its measurements. For that, the proposed system needs to transform their positioning information into grid coordinates, and incorporate them as "votes" with different weight based on their precision.

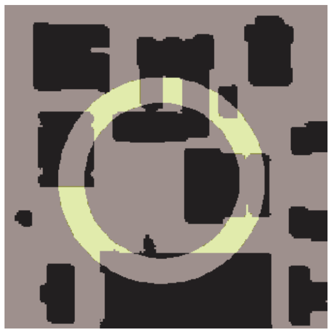


Figure 2: Example of the usage of an outdoors map to filter out the regions where a host cannot be located.

The above algorithm can be easily extended to handle mobility. Depending on its speed, a host can tune the frequency it runs the algorithm. In addition to the positioning information it collects from other hosts, it can also "vote" for a region based on its mobility pattern and speed. For example, it can compute an upper bound of the traveled distance and filter out all of the cells in the grid located outside of that range. In a different setting, such as a freeway, a car may not be allowed to make a U-turn. This can impose an application-dependent constraint in the system. Thus, if a car is located in the lanes traveling south, grid cells lying in the lanes traveling north can be filtered out, if there is no overpass nearby on the map.

If no landmarks or other external information is available to provide the first votes, three hosts can be elected to establish a relative coordinate system using the same approach as in [4]. The elected hosts will subsequently acquire a location in their coordinate system and start acting as legitimate landmarks broadcasting those coordinates and providing positioning information and votes.

The privacy requirement is a major issue in location-sensing. In the centralized architecture, a trustee server can provide location-sensing to authenticated devices. Similarly, the distributed architecture assumes cooperative devices that can authenticate each other before participating in the location-sensing mechanism. Due to the space limitations, we refer the reader to [6] for a discussion on the privacy issues and architecture design.

3 Performance analysis

We run both simulations and actual measurements to evaluate the performance of the system. The *accuracy level* is the main metric used in the performance analysis of our system. We define the accuracy level to be (α, ϵ) if the α percentage of hosts can estimate their location with an error of *at most* ϵ units. We are interested in investigating the impact of the density of landmarks in the area, maximum transmission range, *grid resolution* (i.e., total number of cells), *degree of connectivity* of the network, and *range error* (i.e., the error in the estimation of the distance between two hosts) on the performance of the system. In all measurements that assume a percentage of landmarks (such as x%) we performed 100 runs of the algorithm. In each run, x% of all hosts were randomly selected to be landmarks. Sections 3.1 and 3.2 discuss the performance results via simulations and actual measurements, respectively.

3.1 Measurements via simulations

We consider a terrain of 100x100 square units in size with 100 nodes randomly placed in the terrain. The degree of connectivity, error in determining the distance between hosts, and percentage of landmarks among the hosts are some of the parameters whose affect on our algorithm we will present in this section. In the following figures, we measure the "location error" as a fraction of the maximum transmission range. Also, we use a range error of 5% unless otherwise noted.

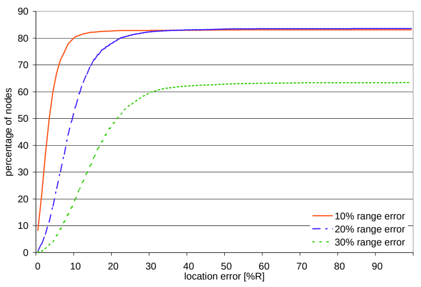


Figure 3: Accuracy level as a function of the range error.

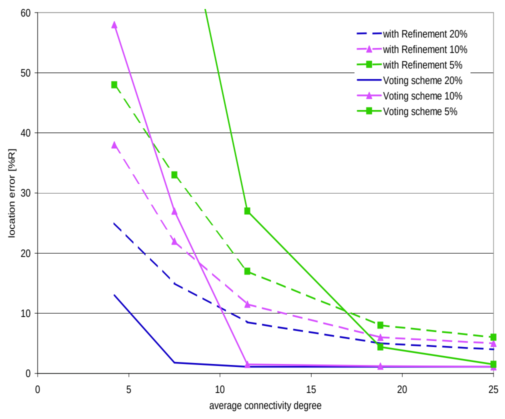
Figure 3 shows the percentage of nodes whose location was computed with at most some error, for range errors of 10%, 20%, and 30% of the maximum transmission range. Figure 4 (b) illustrates the impact of the range error on the accuracy of location sensing. It presents the average error in location sensing when the ranging information has a maximum error ranging from 5% to 50% of the maximum transmission range. Figure 4 also includes the results of the algorithm presented in [15], both with (i.e., "with Refinement") and without the second phase of refinement (i.e., "Hop-TERRAIN").

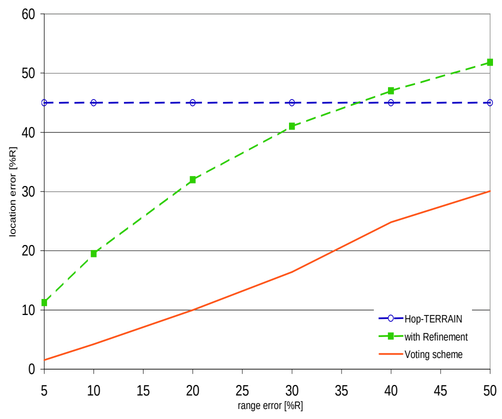
Figure 4 (a) shows the average location error for different average degrees of connectivity and number of landmarks. Figure 4 (a) also includes the results of [15] after their "refinement" step. Though our results are worse for networks with low degree of connectivity, or very few landmarks, we believe that 10% landmarks is a fair estimate of a widely deployed IEEE 802.11 network. We used the traces from an extensive empirical study of the UNC campus wireless infrastructure that includes over 230 APs [5]. Using this trace we computed the maximum number of hosts that were associated with each AP at any given time and then computer the average of those maximums. Our results show a worst case AP to mobile host average ratio of approximately 1:10.

3.2 Performance using actual measurements

We also perform actual measurements to evaluate the performance of the system [1] in both indoor and outdoor environments. Another team in our group [10] performed an empirical study on signal strength using the wireless infrastructure in the third floor of the Department of Computer Science at UNC. We used their results in our actual measurements to provide a more realistic estimation of the distance information between two (non-AP) hosts. For the part of our experiments that involved actual measurements of signal strength, we used the existing infrastructure in the third floor of the Computer Science department of UNC which is composed of two APs placed as shown in Figure 5 (a). The APs participated in the location-sensing system and acted as landmarks. We took measurements in two sessions several days apart to avoid time scale dependencies that would result in optimistic accuracy estimates

[1]We have implemented the system in C++ on Windows. More information about the measurements and the implementation of the location sensing algorithm can be found in [6].





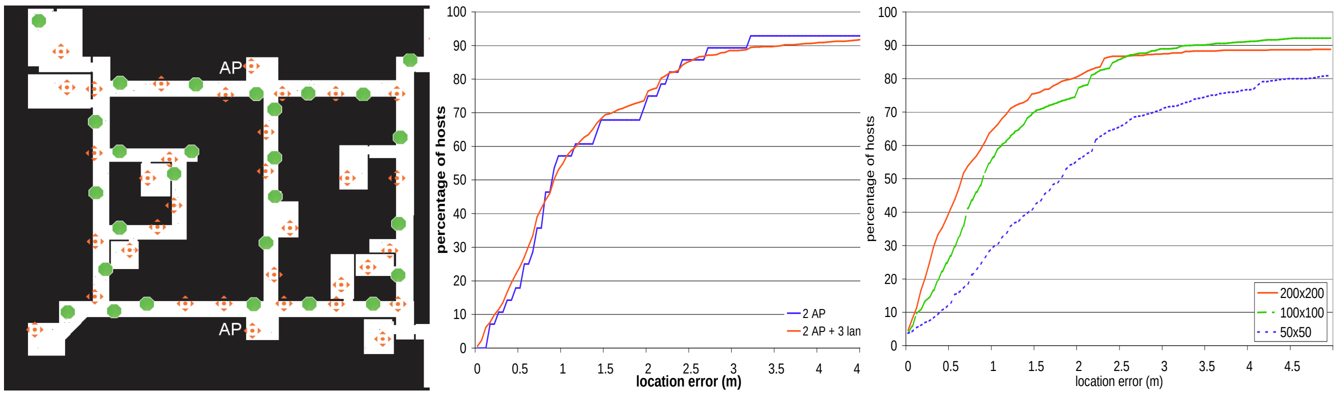
(a) (b)

Figure 4: Performance comparison of our system (L%) vs. "Hop-TERRAIN" and "with Refinement" (L%) from [15] regarding the impact of the average degree of connectivity, landmark percentage (L%), and range error on the location error. The location error percentage and range error are with respect to the maximum transmission range (R).

as suggested by [1].

In the first session, we created signal strength maps. The signal map is a map in the form of a grid in which each cell contains the signal strength information for each AP in the corresponding location. In the first session we took signal strength measurements for both APs at 36 different locations marked on Figure 5 (a). Each measurement consists of the minimum, maximum, and average measured signal strength. To compute the average, we recorded the beacon messages broadcasted by the AP until our average became stable with almost 6 bits of accuracy. The sampling at each location lasted a few seconds. We artificially populated the rest of the signal strength map by using cubic interpolation of the actual samples. For the second session we consider 28 positions different from the ones of the first session. For each position, we computed the average signal strength with a smaller accuracy of approximately 5 bits. Each of these 28 positions represents the position of a host that runs our algorithm during the second session. Each of these hosts has a copy of the signal maps computed during the first session. At the second session (of the measurements), each of these hosts collects signal strength information from the IEEE802.11 beacon packets of each AP. It compares this signal strength information against its signal maps. More specifically, if a cell's corresponding area in the signal map had a value different from the measured value by less than 10% of the difference of the corresponding maximum and minimum measured values in the signal map, the cell accumulated a vote on behalf of the given AP.

In Figure 5 (b) the step-like line shows the accuracy level in a setting with no landmarks. More specifically, the first host computes its position based only on information from the AP signal strength map; the second host derives its position based on the information from the AP signal strength map *and* the distance estimation between the local host and the first solved host and so on. The curvy line corresponds to a setting with three landmarks in addition to the two APs. The step-like line corresponds to a deterministic setting, because hosts are selected in a deterministic order. Therefore, its results are based on one test run. For the curvy line, we run 100 tests and average the results. The three landmarks are not included in the percentage of hosts that estimate their location in Figure 5. That is, the percentage corresponds to the remaining 25 hosts in the terrain.



(a) (b) (c)

Figure 5: Impact of the number of landmarks and grid resolution on the accuracy level in the actual measurements. Figure (a) flooplan of third floor of the Department of Computer Science at UNC. It includes the locations of the two APs. It shows the distribution of the positions at which the signal strength was measured for the two sessions (solid disk and diamond-shape). Figure (b) compares a setting with two APs and no other landmarks vs. one with two APs and three additional landmarks. Figure (c) shows the impact of grid resolution.

We compare the results from using different grid resolutions, namely 50x50, 100x100, and 200x200. In these experiments we consider an AP and three landmarks. These three landmarks have been excluded from the results in Figure 5 (c). The area of the map is 44.9 x 38.3 m^2, so the 50x50 grid can introduce an error of around one meter due to its resolution. The results correspond to an average of 100 tests.

The grid size of the voting algorithm affects the accuracy level. Specifically, the finer the grid, the higher the accuracy, but also higher the memory and CPU power requirements are. In general, in the distributed approach, nodes do not need to choose the same grid size. For example, a PDA with limited capabilities can trade the location-sensing accuracy for the computational complexity, whereas a laptop computer can use a finer resolution grid. Figure 5 (c) illustrates the accuracy level for a grid resolution of 50x50, 100x100, and 200x200 cells.

Our oudoor experiments used a similar methodology as the indoors. Six APs were measured at 24 locations and 16 hosts participated in the experiment.

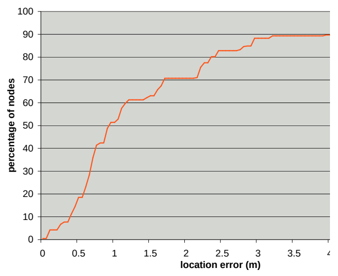


Figure 6: Accuracy level as a function of the range error in an actual outdoor experiment using the system.

4 Related work

Significant work has been published in recent years in the area of location sensing using RF signals. RADAR [3] uses maps of signal strength similar to the ones we exploit is Section 3. RADAR maintains a database of $(x, y, Z, SS_i \quad i = 1, \quad \ldots, n \quad)$ values, where x, y, and z are the physical coordinates of the training sample and SS_i is the signal strength measurements from the i-th AP. Each measured signal strength vector is then compared against the database and the coordinates of the best matches are averaged to give the solution. The authors report that 90% of the time their hosts can be located with at most 6 meters of error with a sampling density of 1 sample every 13.9 and 19.1 square meters of their testbed areas using the signal from three and five APs respectively. Though our work can take advantage of signal map information, as demonstrated is Section 3, the fundamental algorithm is significantly different and emphasizes on cooperation between the hosts, rather than individual host effort. Our results show that with a sampling density of 1 sample every 47.5 square meters and using only two APs we can predict the location of a host 90% of the time with at most 3 meters of error.

Ladd *et al.* [9] propose a substantially different algorithm than RADAR or our solution based on a two step process. In the first step a host uses a probabilistic model to compute the conditional probability of its location in a number of different locations based on the received signal strength from 9 APs. The second step of their algorithm exploits the limited maximum speed of mobile users to refine the results (of the first step) and reject solutions with significant change in the location of the mobile host. Our system does not consider host mobility explicitly. However, our algorithm can be easily extended to incorporate a limit of the location distance of a host between two consecutive runs of the algorithm (as discussed in Section 3.2). Ladd *et al.* report that 83% and 77% of the time, hosts can predict their location within 1.5 meters assuming an infrastructure of nine APs, when the second step is used and not, respectively. Figure 5 (b) shows that our system can predict the location with the same error 70% of the time, but with two APs and cooperation among hosts.

Cricket [13] is another location sensing system that works using quite a different methodology and primarily targets a different kind of applications than our system. The basic idea behind cricket is mounting RF and ultrasound beacons on the walls and ceiling, and using the fact that the two types of signals have different propagation speeds. Our work is significantly different from theirs in that Cricket requires investing in extensive infrastructure of specialized hardware used only for this purpose. Our system on the other hand can operate with limited or no infrastructure. Furthermore, the 802.11 APs that we consider infrastructure are already widely deployed and they do not exhaust their usefulness in providing service to our system. On the other hand, Cricket provides a location sensing granularity of just 4x4 feet. Though our system can only provide significantly worse location sensing accuracy, it is very easy and inexpensive to deploy and maintain. For example, if Cricket sensors were to cover the area of our testbed, it would require deploying and maintaining more that 260 Cricket sensors.

Niculescu and Badri Nath [12] designed and evaluated a cooperative location-sensing system that operates in an ad hoc network. It uses primarily angle estimations and assumes specialized hardware that allows a host to calculate the angle between two hosts. This can be done through antenna arrays or ultrasound receivers. Like in our system, hosts gather data, compute their solutions, and propagate them throughout the network. It is not easy to compare their results with ours due to the different metric used, namely distances vs. angles. Niculescu's previous work [11] also shares the same cooperative spirit among hosts with our work. They introduce an algorithm on location sensing that works on simple geometric principles of Eucledian geometry concerning triangles and quadrilaterals. Like in our algorithm, the information of the landmark locations is slowly propagated towards the nodes that are further away while at the same time closer nodes enrich this information by determining their own location. Three variations of this algorithm are presented: "DV-hop", "DV-distance", and "Eucledian". Though our system is closer to "DV-distance" in that it uses signal strength information to estimate the distance between hosts, its performance more closely resembles the performance of the "Eucledian" scheme, but with always 100% solved hosts. More specifically, we outperform all variations of Niculescu's algorithm for 20% or more landmarks, average degree of connectivity 7.5 and distance approximation error of 5%. However, Niculescu's "DV-hop" and "DV-distance" variations outperform our algorithm when the number of landmarks becomes 5% and the "DC-distance" produces equaly good results at such low percentage of landmarks. For 10% landmarks, our results are almost equally matched. For example, using 5% landmarks Niculescu's algorithms have an error of approximately 25%, 45% and 55% of the transmission range (for the "DV-hop", "DV-distance" and "Eucledian" algorithms) compared to our error of 85%. However, when the number of landmarks increases to 20%, our error falls to 1.8%, while Niculescu's algorithms perform at approximately 12%, 25% and 12%. Another related work is the one by Capkun *et al.* [4]. They presented an algorithm for location sensing in ad hoc networks without any information available from landmarks or GPS. However, we cannot compare Capkun's results with ours, because they only evaluate the tradeoffs among internal parameters of their system without any measurements on the location error.

The research done by Saverese *et al.* [15] is the closest to our work. They describe a distributed algorithm that determines the position of nodes in an ad hoc network in two phases, namely the startup and refinement phase. In the startup phase, landmarks broadcast their location among all nodes in the network and hosts estimate their position by triangulation. These estimations are rough approximations and are improved in the second phase of their algorithm. The refinement phase proceeds in iterations. At each iteration, each host broadcasts its position estimate, receives the positions and corresponding range estimates from its neighbors, and computes a least square triangulation solution to determine its new position. After a number of iterations, when the position update becomes small the refinement phase stops and reports the final position. Though this work presents many similarities to ours, there are alse several very distinct differences. First, we use a grid to solve the location problem instead of a numerical method. Second, our communication overhead is smaller since we avoid the flooding done in the first phase of Saverese's system by relaying landmark position through the regular exchange of information regarding host information (i.e. the corresponding messages that Sevarese's system exchanges during the second phase). As shown in Figure 5 (b) our algorithm can tolerate initial distance measurement errors significantly better than Sevarese's and performs better in networks with 10% or more landmarks and a connectivity degree of at least 8. However, Saverese's method can cope better with networks of very small connectivity (such as 4) or fewer landmarks (less than 5%).

5 Conclusion and future work

We presented a robust scalable location-sensing system. Wireless devices running this system can cooperate and share positioning information to improve their position estimations. It uses a novel voting algorithm and grid-based representation of the environment to determine the location of a given host. The main advantages of the system is that it can provide a reasonable accuracy level without the need of additional hardware, training, and infrastructure. It can work in both indoor and outdoor environments and can be easily extended to incorporate external information (such as GPS, application-dependent semantics, floor plan, signal strength maps) to improve the location estimation. In addition, depending on the characteristics of the environment, it can operate in a distributed, centralized, or hybrid fashion.

We plan to model this cooperative location-sensing algorithm. We would like to investigate analytically the dynamics among the topological properties of the network, wireless range, density of devices and infrastructure, cooperation paradigms, and movement pattern and their impact on the accuracy level. This will also allow us to more systematically evaluate the communication overhead, convergence issues, and the role of backtracking for improving the accuracy. The algorithm exhibits nice scaling properties. In the case of large regions or high density of devices and landmarks, several groups of devices can create a separate network and apply the location-sensing algorithm independently. The above analytical study can provide further insight on how this grouping can take place. In addition, we would like to provide further guidelines for how the system can automatically tune the ST and LECT thresholds for increased accuracy. We are also interested in experimenting with mobile hosts. We plan to extend the system so that data collected during the system operation (potentially from different peers) can dynamically refine the initial training. In this self-learning process, there would be cooperation among peers but limited user intervention. Advances in the mobile robotics area may provide useful insight towards this goal.

References

[1] A Probabilistic Approach to WLAN User Location Estimation. http://www.wlan01.wpi.edu/proceedings/wlan18d.pdf.

[2] The Wireless Island of Patmos, Greece. http://www.12net.gr/802.11.html.

[3] Paramvir ("Victor") Bahl and Venkata Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Tel Aviv, Israel, March 2000.

[4] S. Capkun, M. Hamdi, and J. P. Hubaux. GPS-Free Positioning in Mobile Ad-Hoc Networks. In *Proceedings of HICSS*, Hawaii, January 2001.

[5] Francisco Chinchilla and Maria Papadopouli. Analysis of wireless information locality and association patterns in a campus. Technical report TR03-027, Department of Computer Science, The University of North Carolina at Chapel Hill, Chapel Hill, July 2003.

[6] Charalampos Fretzagias and Maria Papadopouli. Cooperative Location Sensing for Wireless Networks. http://www.cs.unc.edu/˜maria/projects/CLS/.

[7] Jeffrey Hightower and Gaetano Borriello. A Survey and Taxonomy of Location Sensing Systems for Ubiquitous Computing. Technical report.

[8] Jeffrey Hightower, Roy Want, and Gaetano Borriello. Spoton: An indoor 3d location sensing technology based on rf signal strength,. Technical report.

[9] A. M. Ladd, K. E. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. S. Wallach. Robotics-based location sensing using wireless ethernet. In *Proceedings of the Eight ACM International Conference on Mobile Computing and Netwrking (MOBICOM 2002)*, Atlanta, GE, September 2002. ACM Press.

[10] Josh Levy, Alok Shriram, Travis Sparks, and Karl Gyllstrom. Location Sensing with IEEE802.11. http://www.cs.unc.edu/˜gyllstro/290-086/fi nal-project/wireless-single.pdf.

[11] Drago Niculescu and Badri Nath. Ad Hoc Positioning System (APS). In *Proceedings of the IEEE Conference on Global Communications (GLOBECOM)*, San Antonio, November 2001.

[12] Dragos Niculescu and Badri Nath. Ad Hoc Positioning System (APS) using AoA. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, San Francisco,CA, April 2003.

[13] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 32–43, Boston, Massachusetts, USA, August 2000.

[14] Nissanka B. Priyantha, Allen K. L. Miu, Hari Balakrishnan, and Seth Teller. The cricket compass for context-aware mobile applications. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–14, Rome, Italy, July 2001.

[15] Chris Savarese, Jan Rabaey, and Koen Langendoen. Robust positioning algorithms for distributed ad-hoc wirleess sensor networks. In *Proc. of Usenix Annual Technical Conference*, June 2002.

[16] NYC wireless. http://www.nycwireless.net.