Cooperative Location-Sensing for Wireless Networks

Charalampos Fretzagias
Department of Computer Science
University of North Carolina at Chapel Hill
fretzagi@cs.unc.edu

Maria Papadopouli
Department of Computer Science
University of North Carolina at Chapel Hill
maria@cs.unc.edu

Abstract

We present the Cooperative Location-sensing system (CLS), an adaptive location-sensing system that enables devices to estimate their position in a self-organizing manner without the need for an extensive infrastructure or training. Hosts cooperate and share positioning information. CLS uses a grid representation that allows an easy incorporation of external information to improve the accuracy of the position estimation. We evaluated the performance of CLS via simulation and investigated the impact of the density of landmarks, degree of connectivity, range error, and grid resolution on the accuracy. We found that the average error is less than 2% of the transmission range, when used in a terrain with 20% of the hosts to be landmarks, average network connectivity above 7, and distance estimation error equal to 5% 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 a 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 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 device to compute a route that will then guide the user. However, the GPS does not work everywhere. Typically, the technology 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 bounce off nearby objects, making localization difficult or impossible. Moreover, GPS is not a cost-effective or practical option in environments with high accuracy requirements or small and energy-constrained devices. Several alternative location-sensing approaches have been developed and can be differentiated based on their dependency on an infrastructure, accuracy and scale requirements, cost, signal modalities (radio, ultrasonic, vision, and infrared), and location-determination techniques. For example, location can be determined by estimating the distance of the device from landmarks (i.e., devices with known position or equipped with a location-sensing system) and applying multilateration. The distance estimation may rely on time of arrival, angle of arrival, signal strength, or signal structure measurements. Other location-determination techniques use scene analysis. Hightower and Borriello [11] provide a taxonomy of location-sensing systems.

Recently, IEEE 802.11 networks have become widely available in universities and corporations to provide wireless Internet access. Such networks are also increasingly deployed in airports, hospitals, shopping centers, and other public areas [3, 4]. The cost and complexity advantages of using IEEE 802.11 for both communication and positioning make it an attractive choice. For example, the Radar [5] and the robotics-based location-sensing system in [13] use an infrastructure of IEEE 802.11 access points (APs). IEEE 802.11 cards measure the signal strength from those APs. Based on data gathered by wireless cards at various predefined checkpoints of an indoor environment, they build a signal strength map of the environment. They locate an wireless device by using this map and applying a probabilistic inference of the position. Other approaches, such as Bat [9] and Cricket [17], use an extensive infrastructure deployed indoors and combine ultrasonic and radio signal measurements to achieve higher accuracy. For example, in Bat, a radio signal from a central controller triggers an active tag at a known time. When triggered, the tag emits an ultrasonic pulse, which is received by a matrix of receivers placed at known locations in the ceiling. By measuring the time delays between emission and reception of pulses, they gain estimates of the distance between the tag (on the mobile device) and each receiver in its range. Using a multilateration algorithm, they convert the distance measurements to a location in the space.

In many situations, due to environmental, cost, maintenance, and regulatory barriers, the deployment of a dense infrastructure for location sensing is not feasible. There are several research efforts that addressed the challenges of location-sensing in ad hoc (infrastructureless) or sensor networks and developed location sensing systems [18, 15, 12, 6, 20, 19, 10]. They assume cooperative devices that operate in a self-organizing manner. Hosts use positioning estimates from landmarks and neighboring hosts to iteratively computer their location. These systems vary on their configuration (type, density, and transmission range of devices and landmarks), communication protocol, range approximation, and position estimation methods.

CLS also builds on the idea of collaborative location-sensing. In CLS, hosts estimate their distance from their neighboring peers. This can take place with any distance-estimation method available (e.g., using signal strength and a signal attenuation model). They refine their estimations iteratively as they incorporate new positioning information from their neighbors. The main difference of CLS from other collaborative location-sensing methods in ad hoc or sensor networks [18, 15, 20, 6, 12] is the use of a grid-based representation of the terrain and the voting algorithm. In general, the distance estimations are often inaccurate; for example, due to multipath, fading, and shadowing effects, it is difficult to obtain a consistent model of the signal attenuation as a function of distance. These *range* errors are propagated through the positioning messages in the network. The voting algorithm accumulates and assesses the positioning information received from other hosts and through a selection and averaging process attempts to mask such errors. In pervasive computing, the availability of presence, occupancy, floorplan and signal maps of the environment is critical. This motivated further the use of the grid representation and voting algorithm that allow the system to easily incorporate such additional information and refine its position estimates. Sections 3 and 4 compare CLS with several location-sensing systems in detail.

The main contribution of this paper is the development and evaluation of CLS, a collaborative location-sensing system, that uses a novel voting scheme to incorporate positioning information from hosts in the network and enable them to estimate their position in a self-organizing and adaptive manner. CLS can adapt based on the availability of infrastructure, training data, energy and computational constraints, and accuracy and precision requirements. For example, it can trade some accuracy in the position estimates for lower computational complexity by reducing the grid size. To reduce the communication and computational overhead, it can also use a centralized or hybrid architecture. We evaluate CLS with simulations and investigate the impact of the density of landmarks, degree of connectivity, error in the distance measurements, and grid resolution on the accuracy of CLS. We show how CLS can be extended to take advantage of an wireless infrastructure and training data and discuss its performance.

The dynamic characteristics of the environment and nature of the signal propagation impose important challenges on the design of a scalable, easily deployable, and computationally inexpensive location-sensing system. The following design characteristics shape our vision for CLS. It should be

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.

Section 2 presents an overview of CLS, its communication protocol, voting algorithm, and different architecture designs. Section 3 analyzes its performance via simulations. Section 4 discusses related work on location-sensing using an infrastructure or in ad hoc networks. Finally, in Section 5, we summarize our main conclusions and future work plans.

2. Overview of CLS

The objective of the CLS is to determine the position of devices in a self-organizing manner and without the need of an extensive infrastructure or training. The CLS system consists of a communication protocol and a voting process. The communication protocol disseminates positioning and distance estimates among hosts in the network. A host uses these position and distance estimates to compute its position in the voting process.

2.0.1. Active and passive hosts and CLS servers All the hosts in the network participate in the communication protocol, whereas only the computationally powerful hosts run the voting process and compute their location. Depending on whether or not a host computes its own location, it is called *active* or *passive*. We consider three architectures, namely, the distributed, centralized, and hybrid. In the distributed architecture, all hosts are active whereas in the centralized all hosts are passive. In the hybrid architecture, active and passive hosts are present. In the centralized and hybrid architectures, CLS servers compute the position of their local passive hosts. Passive hosts can discover their local CLS server through a simple service discovery protocol (e.g., broadcasting queries for available CLS servers in close proximity [16]). CLS servers can be either part of an infrastructure available to CLS or active hosts that cooperate as CLS servers.

2.0.2. CLS run The CLS runs iteratively at each host. For each iteration, it considers a snapshot of the network and assumes that all hosts are stationary during that run. An active host reports its position as soon as it computes it during a run. A CLS server computes the positions of its local passive hosts (i.e., reference hosts) in the network. We refer to the duration of an iteration as a *CLS run* or *run*. The exchange of CLS messages and the voting process take place concurrently during a run.

2.0.3. Grid We use a *grid-based* representation of the terrain. Each host maintains a regularly-spaced grid of cells. At the beginning of a CLS run, hosts initialize their grid and during the voting process, hosts cast votes on the cells. Each host is configured with a *voting weight*, a constant, that depends on the confidence of the host about its position estimation. We assume that landmarks have higher voting weight than hosts that compute their position using CLS. The value of a cell in the grid is the sum of the voting weights of all those hosts that casted votes for that cell during a CLS run. The higher the value of a cell is, the more hosts agree that this is a likely position of that host. A cell is represented using the local grid coordinate system. We assume that all hosts in the terrain share a common (global) coordinate system used to represent their position. Each host knows its grid resolution and can transform these position coordinates to coordinates of its local grid. The grid size or resolution (determined by the cell area) does not need to be the same for all hosts.

2.1. Communication protocol

2.1.1. CLS beacon and update messages A *positioning entry* of a host consists of several fields, namely, its host id, maximum wireless transmission range, position, and voting weight. The position field may be empty or include the position and time it was configured or computed. We assume that each host is configured with its maximum wireless transmission range.

A CLS run starts with a neighborhood discovery protocol that enables hosts to learn about their *one-hop* neighbors (i.e., hosts within their range). At the beginning of a run, a host broadcasts a *CLS beacon* with its own positioning information. It is a *single-hop* broadcast and only hosts within the range of the sender may receive its beacon. Hosts that receive this beacon are required to respond with their own positioning information to announce their presence.

A recipient of such message can estimate its distance from the sender with an available distance estimation method (e.g., based on signal strength). For that, CLS assumes a radio propagation model that converts a signal strength value to a distance interval $(d - \epsilon, d + \epsilon)$, where d is the mean distance estimate and ϵ the range error. A host combines this distance estimate with the position information of the sender. This set of information that a host maintains for a neighboring host is defined as the *CLS entry*. For example, consider host m that receives a CLS beacon from host n that has id id_n, position p_n computed at time t_n, transmission range R_n, and voting weight w_n. Host m computes its distance from n to be the interval $(d^l_{m,n}, d^u_{m,n})$ and generates a CLS entry for n set to $(id_n, p_n, t_n, R_n, w_n, (d^l_{m,n}, d^u_{m,n}))$.

When a passive host generates a *new CLS entry*, it forwards it to the CLS server, while an active host disseminates it in a controlled manner in the network. By controlled dissemination, we mean that active hosts only rebroadcast the updated or new CLS entries to prevent infinite loops and repeated broadcasts of the same or stale information. Stale entries are distinguished based on the position field and Time-To-Live (TTL) value. The dissemination of such message is constrained by its TTL value, because its impact on the location estimation diminishes as a receiver moves further away from the original sender. To reduce the communication overhead, a host may aggregate new positioning entries to a single message. We refer to these messages as *CLS updates* (or just updates). CLS updates enable active hosts to learn about other hosts in the terrain, some of which are more than one-hop away.

When a CLS server computes the position of a host, it forwards an update with the position estimate to the reference host. When a host either computes its position or receives it from the server, it broadcasts a new beacon. Section 2.2 discusses how this information is used in the voting process of other hosts.

2.1.2. CLS table An active host maintains a table with all the received CLS entries. A CLS server maintains similar table for each host that it tries to position (i.e., reference host). As mentioned earlier, passive hosts forward new CLS entries to their server. This table is initialized at the beginning of a run and updated when the local host gathers a non-stale CLS update. Note that it contains entries not only for hosts that are within the range of the reference host but also for hosts that the reference host has learned about through updates (from its neighbors). For example, assume that a (reference) host, e.g., host u, has not received any beacon or update *directly* from a given host (e.g., host C) but has a positioning entry about it (sent from another host, host A). CLS infers that C is likely not in the wireless range of u. The distance interval between u and C is set to (R_C, ∞), where R_C is C's maximum wireless.

2.2. Voting process

We represent as G_m the grid that a host m maintains during a run and $v(x, y)$ the value of the cell (x, y). The value of a cell reflects the agreement of other hosts in the network that this cell is a likely position of the local host. A CLS server maintains similar grid for each of its reference hosts. The voting process takes place as the local host (or CLS server) receives CLS update messages and builds the table.

We define the *position estimation* of host m from host n, $P_{m,n}$, to be the set of cells in G_m that are likely positions of host m given its CLS entry about host n (i.e., the position of host n, its wireless range R_n, voting weight w_n, and their estimated distance interval $(d^l_{m,n}, d^u_{m,n})$). For the voting process, a host transforms the global coordinates of all the (acquired) position information to coordinates of its local grid.

In the case that the two hosts (e.g., hosts m and n) are one-hop neighbors, the set $P_{m,n}$ is equal to

$$P_{m,n} = \{(x, y) \in G_m,\, d^l_{m,n} \leq \sqrt{(x_n - x)^2 + (y_n - y)^2} \leq d^u_{m,n}\}.$$

If host m only learns about n indirectly, its position from host n is defined as

$$P_{m,n} = \{(x, y) \in G_m,\, R_n \leq \sqrt{(x_n - x)^2 + (y_n - y)^2}\}.$$

At each run, each host (e.g., host h) gathers updates and attempts to compute its own location performing the following steps:

Step 1. At the beginning of a run, initialize the value of the cells in the grid G_h by assigning zero to them:

$$\forall (x, y) \in G_h,\, v(x, y) = 0 \tag{1}$$

Step 2. For each CLS entry about a host with known or computed location (e.g., host k), perform the following steps:

1. Compute $P_{h,k}$.
2. Increase the value of the cells in $P_{h,k}$ by w_k, where w_k is the *voting weight* of host k:

$$\forall (x, y) \in P_{h,k},\; v(x, y) = v(x, y) + w_k \tag{2}$$

Step 3. The set of cells in the grid with maximal values defines a possible region where host h may be located. Landmarks and hosts that computed their own location earlier in a run determine to some extent the location of other hosts and their estimation errors are propagated in the network. To reduce the impact of these errors, we apply the following heuristic. During the voting process in a CLS run, a set of cells has to satisfy the following two conditions in order to be an acceptable solution:

- The number of votes in each cell of the potential solution must be above a threshold. We refer to this threshold as the *voting 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 and LECT its precision. A high ST value reduces the error propagation throughout the network but delays the positioning estimation. 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. The values of ST and LECT also depend on network characteristics such as the density of hosts, ratio of landmarks and hosts with unknown location, and range errors.

Step 4. The local host casts votes from the newly generated CLS entries of its table until the two conditions are satisfied. That is, if the above two conditions fail, the voting process returns to Step 2. The local host may also adaptively relax these thresholds. In Section 3, we describe the heuristic that we use. When the host finds an acceptable set of cells with maximal values, it computes its centroid and terminates the voting process. The centroid's coordinates are the average of the respective coordinates of the cells with maximal values. The host transforms the coordinates of this centroid to coordinates of the shared (global) coordinate system. This centroid becomes the position of host h for this CLS run.

2.2.1. Example of voting process

Let us consider hosts A, B, and C with known or estimated position and host u that currently tries to position itself. We discuss this example from the perspective of host u, after it received beacons and updates from A and B and updated its CLS table.

Figure 1 shows the grid of host u, after it accumulated the votes from A, B, and C. On the grid, we have superimposed the positions of A, B, and C, the distance estimate between A and u ($(d_{u,A} - \epsilon, d_{u,A} + \epsilon)$), and the transmission range of C, R_C. Each cell is marked with its value, given that A, B, and C's voting weight is one and are the only hosts that have voted so far. The darker an area is, the more voting weight has been accumulated on the corresponding grid cells.

In this example, we assume that u has not received any beacon or update from C but learns about C from either A or B's update. It then infers that its position is likely to be outside of the range of C and casts a vote for the cells in the grey area outside of the range of C. The set of cells with the maximal value defines a possible solution. In this example (so far), this set consists of the two cells with value 3. In order to be accepted, the two conditions of the voting process need to be satisfied. Once a potential solution has been accepted, the algorithm computes the centroid of that set of cells and reports it as the estimated location of the newly solved host. The host does *not* try to refine further its loca-

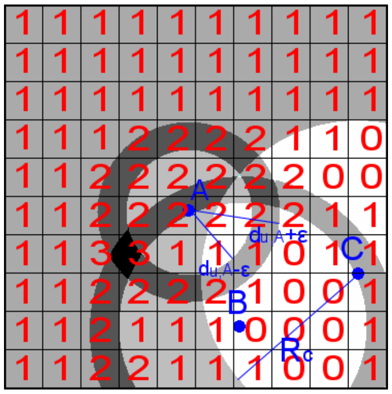

Figure 1. Accumulation of votes on the grid cells of host u (with unknown position) after A, B, and C have casted their votes. A and B contribute with positioning information, whereas u only learns about C from either A or B. The position of hosts A, B, and C, wireless range of C, R_C, and distance interval from u to A are superimposed on the grid.

tion estimation during that run, even when it receives additional positioning information from other hosts.

The centralized approach is easy to implement, has smaller communication overhead, and does not require from hosts to perform any computations apart from the signal strength to distance interval conversion. Specifically, the total communication overhead of CLS during a run consists of the (single-hop) broadcasts of beacon and the forwarding of these messages to the server. Assuming a routing protocol that enables hosts to route messages to the CLS server, this results to $O(d * N)$ messages per run, where d is the average hop distance to the CLS server and N the number of hosts. However, the centralized approach suffers from the single-point failure problem and scales as much as the available computational power of that server for a given region.

2.3. Incorporating external information

[1] CLS 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. The voting process can be easily extended to consider a more sophisticated signal propagation model than the ideal disk model of the example. In Section 3.1, we discuss and evaluate such extension, namely, the use of a signal map of the environment. We introduce a training phase during which this map is built using signal strength information received from the IEEE802.11 infrastructure. During the voting process, a host considers the signal map information in addition to its distance measurements. Active hosts and CLS servers could acquire these maps via a simple resource discovery protocol. We evaluate the extended CLS in Section 3.1.

The voting process can also incorporate occupancy, presence, trajectory, or floorplan information in the map 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 or obstacles are not likely positions for nodes. Harle and Hopper [8] developed a system that can assist in generating such map information by extracting topological and geographical data from an indoor environment. These are collected positional data sensed from personnel movements in an indoor environment using the Bat infrastructure.

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 [6]. The elected hosts subsequently acquire a location in their coordinate system and start acting as legitimate landmarks broadcasting those coordinates and providing positioning information and votes.

3. Performance analysis

We run simulations to evaluate the performance of the system in a stationary environment. The *accuracy level* is the main metric used in the performance analysis of CLS. 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. This percentage considers all nodes in the terrain excluding landmarks. We investigate the impact of the density of landmarks, maximum transmission range, *grid resolution* (i.e., total number of cells), *degree of connectivity* of the network (i.e., average number of hosts, including landmarks, within the range of a host), and *range error* (i.e., distance estimation error) on the performance of the system.

3.0.1. Simulation testbed We consider a terrain of 100x100 square units in size with nodes randomly placed in the terrain. The default number of nodes in the terrain is 100. Landmarks and hosts with unknown position have the same transmission range. Both the location error and range error are specified as percentages of the transmission range. For example, 50% location error means half the transmission range. We assume a range error of 5% unless otherwise noted.

1 The privacy requirement is a critical issue in location-sensing. To enable cooperation among peers and CLS servers, an authentication mechanism is required. Due to the space limitations, we refer the reader to [2] for a discussion on the privacy issues and architecture design.

We set the distance interval between two hosts in the terrain to be $(\max(0, d - \bar{\epsilon} * R), \min(R, d + \bar{\epsilon} * R))$, where d is their actual distance, R the maximum transmission range, and $\bar{\epsilon}$ the range error.

For each setting with a certain percentage of landmarks (such as x%), we performed 100 runs of the algorithm and averaged the results. In each run, x% of all hosts were randomly selected to be landmarks.

The voting weight of a landmark is set to the total number of hosts (including landmarks) in the terrain and the voting weight of a node to one. The grid size is indicated as AxA and the default is 100x100.

3.0.2. Impact of range error Figure 2 shows the accuracy level for range errors of 10%, 20%, and 30%. In these settings, the network has an average degree of connectivity equal to 10 (including landmarks) and 10% of hosts are landmarks.

Figure 3 illustrates the impact of the range error on the accuracy of CLS ("Voting Scheme"). In the following paragraphs, we discuss the other two schemes, namely "Hop-TERRAIN" and "with Refinement"[18] and compare them with CLS. The configuration setting in Figure 3 corresponds to a network with 10% landmarks and average connectivity degree of 12. It presents the average error in location sensing (for all hosts with unknown position) when the range error varies from 5% to 50% of the maximum transmission range. For example, for a high range error (equal to half the transmission range), the average position error is 30% (of the transmission range). As mentioned earlier, the percentage of hosts for the average position error in a setting includes only the hosts with unknown positions (90 hosts in this example).

3.0.3. Impact of connectivity degree and percentage of landmarks We compared CLS with a distributed location-sensing system for ad hoc networks developed by Saverese *et al.* [18]. Their system operates in two phases, namely the startup ("Hop-TERRAIN") and refinement phase ("with Refinement"). In the startup phase, landmarks broadcast their location among all nodes in the network and hosts estimate their position via triangulation. These estimations are rough approximations and are improved in the refinement phase. 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.

The distributed CLS operates in a similar manner as the refinement phase. Their main differences are the use of grid and voting as opposed to solving a system of linear equations and their approach of handling the propagation of erroneous position estimations. Similar to the voting weight in CLS, in their system, each node associates a confidence value with its position estimates. Their system mitigates the error propagation by adjusting this confidence value. For example, a node with few neighbors and landmarks in its range gets a low confidence for its position estimates. In contrast, in CLS, the voting weight of a host remains fixed. However, CLS adjusts the ST and LECT thresholds to deal with poor network conditions and error propagation.

A CLS host sets initial "optimal" values on the ST and LECT thresholds based on the density of landmarks in the terrain. However, after a number of failed attempts to satisfy them, it iteratively relaxes them. More specifically, in this paper, we experiment with the following heuristic: during a run, each local host keeps a counter of the number of times that either one of the two condition tests fails. When

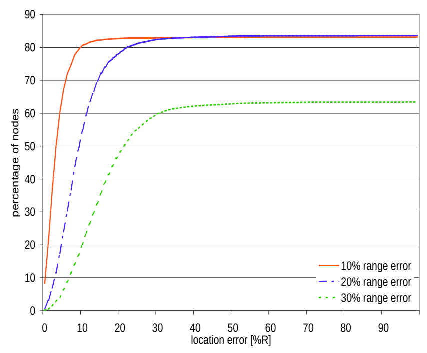

Figure 2. Accuracy level as a function of the range error. 10% of hosts are landmarks.

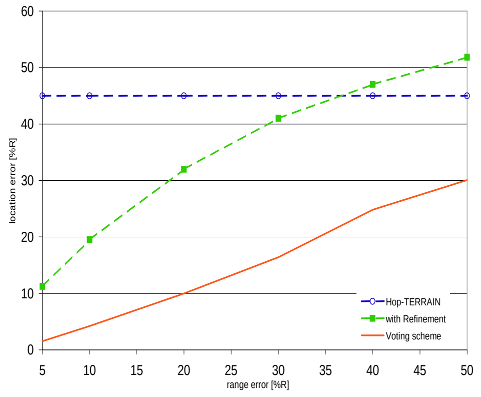

Figure 3. CLS "Voting scheme" vs. "Hop-TERRAIN" and "with Refinement": impact of range error. 10% landmarks and average connectivity degree of 12.

such failure occurs, the host increases the counter and resets the ST and LECT thresholds. Specifically, it updates the ST threshold by dividing the initial required number of votes from non-landmarks (part of the ST threshold) with this counter. Also, it multiplies the initial LECT threshold with this counter. The required number of votes from landmarks is kept fixed to ensure a higher accuracy.

Figure 4 shows the average location error of CLS ("Voting scheme") for different average degrees of connectivity and percentage of landmarks. It also includes the results of [18] after their "refinement" step. Our results are worse for networks with low degree of connectivity or few landmarks but outperform theirs for networks with 10% or more landmarks and a connectivity degree of at least 8 (as in Figure 3 with 10% landmarks and connectivity degree of 12).

We believe that 10% landmarks is a typical estimate of a deployed IEEE 802.11 network where APs act as landmark. We used the traces from an extensive empirical study of the UNC campus wireless infrastructure that includes over 230 APs [7]. We computed the maximum number of hosts associated with each AP (i.e., clients) at any given time and then, the average of those maximum values. This average is 10 clients per AP.

He *et al.* [10] use also signal strength information and a grid representation to accumulate votes from other hosts. However, instead of estimating distances, a host tries to determine whether it resides inside of a triangle defined by three audible landmarks (i.e., landmarks within the range of the host). Moreover, their typical settings are different from ours. In their sensor networks, a sensor has more audible sensors and landmarks than in ours. This imposes more demanding requirements on the sensors. APIT is their main location-sensing system and PIT its "ideal" version that assumes that a host can accurately test if it is inside or outside a triangle of three audible landmarks.

In the next paragraphs, we compare APIT with CLS. In their work, they had also consider irregular radio patterns. However, for this comparison, we only consider their results on regular radio pattern and random placement of hosts in the terrain, since these match our assumptions. Also, for this comparison, we had to repeat the simulations measurements of Figure 4. CLS outperforms APIT even with lower average number of audible landmarks per host. For a given connectivity degree, instead of adjusting the transmission range (as we did in the older simulations), we now vary the total number of hosts in the terrain and keep their range fixed. The size of the terrain remains fixed at 100x100. Specifically, we generated networks with connectivity degree of 4, 8, 12, 18, 25 and a total number of hosts 100, 187, 280, 415, and 580, respectively (for a transmission range fixed to 12.4 units). The new results match the old ones (of Figure 4). In the connectivity degree and total number of hosts, the landmarks are also included. For a connectivity degree of 8 and

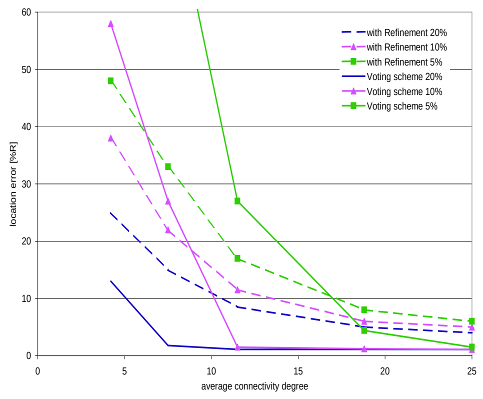

Figure 4. CLS "Voting scheme" vs. "Hop-TERRAIN" and "with Refinement" for various connectivity degrees and landmark %. The range error is 5%.

a total number of landmarks equal to 9, 18, and 37, the location error is 85.2%, 27.3% and 1.84%, respectively. Figure 14A in [10] corresponds to a setting with a density of non-landmark nodes equal to 8. In these settings, landmarks have ten times larger range than non-landmarks (as opposed to our settings that all nodes have the same range). Audible landmarks have much higher impact on the position estimation than non-landmarks or remote landmarks, because they enable hosts to compute useful distance or proximity information. So, although a host in CLS could potentially hear from a remote landmark (through an update), its impact on the position estimation is small.

In a setting with connectivity degree of 12 (7 non-landmark neighbors and 5 audible landmarks per sensor), the position estimation error in APIT is 600% whereas CLS with the same connectivity degree (but with only 0.6 audible landmarks per sensor and 14 landmarks in total) provides an estimation error of 27%. Moreover, in a network with 18 landmarks and non-landmark density equal to 8, CLS results in an estimation error of 27% compared to 60% with 18 audible landmarks per sensor and same non-landmark density in [10]. Actually, in that setting, CLS performs similarly as PIT.

3.1. Extension of CLS using signal maps

We extended CLS to consider the infrastructure of IEEE802.11 access points (APs) and a signal strength map of the environment. We evaluate its performance in an indoor environment.

3.1.1. Training and estimation sessions Our measurements were performed in two phases. In the first phase, i.e., *training session*, we built the signal strength maps from the

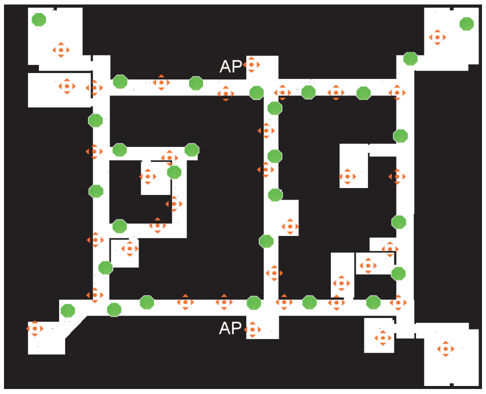

Figure 5. Floorplan of third floor of Sitterson Hall. It shows the two APs, positions at which the signal strength was measured for the training and estimation session (solid disk and diamond-shape, respectively).

APs. For that, we performed indoors empirical measurements in the third floor of Sitterson Hall, where the Department of Computer Science at UNC is located. In the second phase, the *estimation* session, we evaluated the extended CLS using simulations.

The APs of the IEEE 802.11 infrastructure act as landmarks in CLS. At the third floor of the Sitterson Hall, there are two APs (shown in Figure 5). This Figure shows also the positions at which we collected measurements for both the training and estimation sessions. We used a wireless LAN network analyzer software [1] to measure and record the signal strength from APs.

3.1.2. Building signal strength maps In the training session, we created signal strength maps. The *signal map* is a grid-based map in which each cell contains the signal strength information for *each* AP in the corresponding location. In the training session, we took signal strength measurements for both APs at 36 different locations (marked as solid disks in Figure 5). 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.

We denote as SM_i the part of the signal map that corresponds to AP i, and the maximum, minimum, and average signal strength values of a cell c of the map for AP i as $SM_i[c].max$, $SM_i[c].min$, and $SM_i[c].avg$, respectively.

3.1.3. Extending CLS with signal map For the estimation session, we consider 28 positions marked with diamond shape in Figure 5 *different*, from the ones (solid-disk shaped) of the training session. At each of these 28 positions, we collected signal strength measurements from the IEEE 802.11 beacons from each AP and computed an average signal strength value for each AP (with an accuracy of 5bits). We refer to this value as the *measured signal strength value*.

In the estimation session, we evaluated CLS via simulations. For these simulations, the terrain corresponds to the floorplan of the third floor of the Sitterson Hall (Figure 5). Each of these 28 diamond-shape positions represents the position of a "virtual host" that runs CLS in the simulations. Each host has access to a copy of the signal maps computed during the training session and the measured signal strength values. We assume a wireless transmission range of 20m and that each host can communicate with the other hosts in its wireless range. In Figure 5, the average connectivity degree of a host is 6.7 excluding the APs. As described in Section 2, each host maintains a grid that is initialized with zeros at the beginning of the run. We assume that APs act as landmarks and can position themselves on the grid.

At the estimation session, each of these hosts uses the collected signal strength information from the IEEE802.11 beacon packets of each AP and computes an average signal strength for each AP. The step 2(a) of the voting algorithm (described in Section 2) has been modified as follows: The host compares this value against its signal map information (for each AP) and accumulates votes on the cells of its grid as follows: For each cell c, it compares the measured signal strength value $\bar{s}_i$ (from AP i) with the $SM_i[c].min$, $SM_i[c].max$, and $SM_i[c].avg$. If the condition 3 is satisfied, the cell c accumulates a vote "from" AP i.

$$\frac{|\bar{s}_i - SM_i[c].avg|}{SM_i[c].max - SM_i[c].min} \leq 0.1, \tag{3}$$

Figure 6 shows the accuracy level in a setting with no additional landmarks (apart from the two APs). 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 landmarks are not included in the percentage of hosts that estimate their location.

3.1.4. Impact of signal-strength map For that, we contrast the accuracy level provided in the following two settings, namely the one with 10% landmarks (in Figure 2) and one with two APs acting as landmarks (or 6.6% landmarks) and CLS extended with signal strength maps (Figure 6), for different range errors. Even with a smaller number of land-

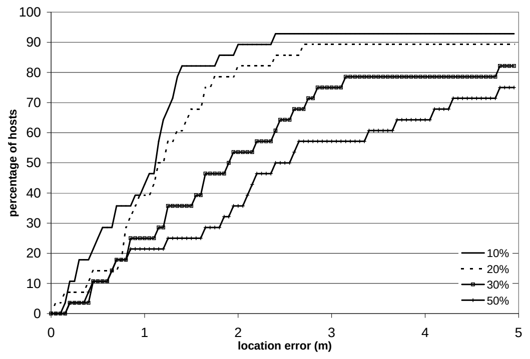

Figure 6. Extended CLS. Impact of the range error on the accuracy level. There are two landmarks in the terrain (the two APs).

marks, the extended CLS guarantees a location error of at most 2m (or 10% of the transmission range) to 53%, 83%, and 89% hosts compared to 20%, 50%, and 80%, for range errors of 30%, 20%, and 10%, respectively. Similarly, the extended CLS guarantees a location error of at most 4m to 78%, 89%, and 93% hosts vs. 50%,78%, and 82%, for range errors of 30%, 20%, and 10%, respectively. Note that the larger the range error, the higher the impact of the signal strength map on the performance of CLS.

3.1.5. Impact of voting from non-landmarks vs. signal strength map We investigate the impact of the voting of non-landmarks on the accuracy level. For that, we define as "no CLS" a scheme in which *only* the landmarks vote. There is no cooperation or voting from non-landmarks in the terrain. Figure 7 shows three "no CLS" schemes with various grid sizes and one CLS scheme with an 100x100 grid size and 20% range error. In all these schemes, the only landmarks are the two APs. They also use the signal-map extension discussed earlier. Note that, CLS without the extension of signal-strength maps and 10% landmarks (Figure 2) for a 20% range error and 100x100 grid size provides an accuracy levels of (50%, 10%) and (78%, 20%), whereas the "no-CLS" (32%, 10%) and (57%, 20%).

The voting from peers has a substantial impact on the accuracy level; for a grid size of 100x100 and range error 20%, the accuracy level of (80%, 2m) provided by the extended CLS falls to (32%, 2m) when there is no voting from peers. Even when the range error increases to 50% (Figure 6), the accuracy level of CLS is higher than "no-CLS"'s, especially for location errors higher than 2.5m. However, if we consider exactly the setting of Figure 5 with two landmarks at the position of the two APs and with all hosts running the plain CLS (i.e., without the signal map extension), the performance suffers; CLS provides an average location error of 136%. The high average location error is due to the very small number of landmarks and relative high range error (20%). For lower range errors (e.g., 5%), in a setting with average connectivity of 4, the plain CLS provides an average error of 57% and 13%, for 10% and 20% landmarks, respectively (Figure 4).

The average location error of the extended CLS that uses 2 APs (Figure 6) is 2.8m, 3.3m, 5.8m, and 6.8m, for range errors 10%, 20%, 30%, and 50%, respectively. The average location error remains similar (for the same range error) when three additional landmarks participate. So, when signal maps are available, a few additional landmarks do not have a substantial impact on the accuracy. Considering a transmission range of 20m, the above location errors are

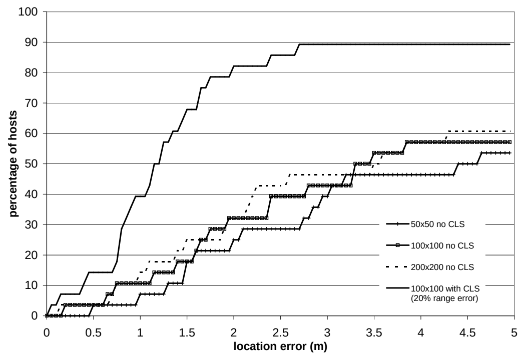

Figure 7. Extended CLS. In "no CLS", only landmarks vote (no voting/cooperation from non-landmarks) and hosts use CLS with signal maps. The "CLS" scheme is CLS extended with signal maps. There are 2 landmarks in the terrain (the APs).

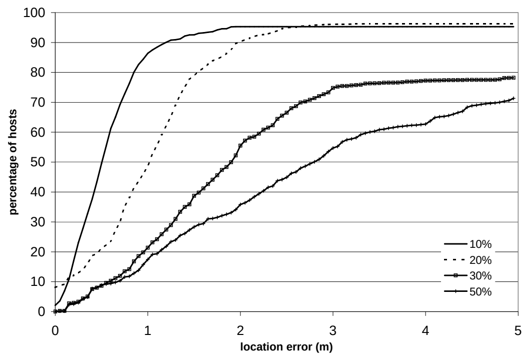

Figure 8. Extended CLS. Impact of the range error on the accuracy level. There are five landmarks in the terrain (two of them are the APs).

14%, 16.5%, 29%, and 34%, for range errors 10%, 20%, 30%, and 50%, respectively. The plain CLS will perform better when there is an increase in the density of peers and landmarks. For example, the average location error is reduced by around 10% in setting with 10% landmarks and connectivity degree 12 (Figure 3).

The grid size of the voting algorithm affects the accuracy level. The finer the grid, the higher the accuracy but also the memory and CPU power requirements. In general, in the distributed approach, hosts do not need to choose the same grid size. For example, a PDA with limited capabilities can trade some accuracy for lower computational complexity, whereas a laptop computer can use a finer resolution grid. Figure 7 illustrates the accuracy level for a grid resolution of 50x50, 100x100, and 200x200 cells. Compared to 200x200 and 50x50, a grid size of 100x100 is a reasonable choice to balance the computational complexity and accuracy requirements. Also, there is only a small gain in the average location error for increased grid sizes (in Figure 7). We found that, it ranges from 6.3m to 6.8m, depending on the grid size.

4. Related work

Significant work has been published in recent years in the area of location sensing using RF signals. RADAR [5] uses maps of signal strength similar to the ones we exploit is Section 3. RADAR maintains a database of $(x, y, z, SS_i\ , i = 1, \ldots, n)$ 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 6m of error with a sampling density of 1 sample every 13.9 and 19.1m^2 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.

Ladd *et al.* [13] propose another location-sensing algorithm that uses the IEEE 802.11 infrastructure. In its first step (of the algorithm), a host uses a probabilistic model to compute the conditional probability of its location for a number of different locations, based on the received signal strength from nine APs. The second step 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. Depending on whether the second step is used or not, 83% and 77% of the time, hosts can predict their location within 1.5m. The extended CLS with five landmarks (two of them are the APs) can provide such location error to 28%-92% hosts, given range errors that vary from 50% to 10% (Figure 8).

Niculescu and Badri Nath [15] 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. 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 and Badri Nath[14] introduce a cooperative location sensing system. Position information of landmarks is propagated towards hosts that are further away while at the same time closer hosts enrich this information by determining their own location. They evaluate three variations, namely the “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, for 20% or more landmarks, average degree of connectivity 7.5 and range error of 5% CLS outperforms the “DV-hop”, “DV-distance”, and “Eucledian”. However, “DV-hop” and “DV-distance” perform better than CLS when the number of landmarks becomes 5%. The “DV-distance” produces equally good results at such low percentage of landmarks. For 10% landmarks, our results are almost equally matched. For example, using 5% landmarks “DV-hop”, “DV-distance”, and “Eucledian” have an error of approximately 45%, 20%, and 55% of the transmission range, respectively compared to the 85% of CLS. However, when the number of landmarks increases to 20%, CLS’s error falls to 1.8%, while the above algorithms perform at approximately 26%, 15%, and 22%.

Another related work is the one by Capkun *et al.* [6]. 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 by Saverese *et al.* [18] and He *et al.* [10] are the closest to CLS. We compare them in more detail in Section 3.

5. Conclusion and future work

We presented a robust adaptive 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 extensive hardware, training, and infrastructure. It can work in both indoor and outdoor environments and can be easily extended to incorporate external information to improve the location estimation.

We plan to model CLS and 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. CLS 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.

There are several types of errors due to radio propagation models and error range, mobility, packet losses, and voting algorithm. These errors are propagated in the network through CLS messages. In this paper, we used heuristics for adjusting the LECT and ST thresholds to control some of these errors. We would like to provide further guidelines on how the system can automatically tune these thresholds for increased accuracy. This problem becomes challenging especially in mobile environments. Advances in the mobile robotics area may provide useful insight towards this goal.

References

[1] Airopeek nx: Wildpackets wireless lan analyzer (demo version). http://www.wildpackets.com/products/airopeek_nx.

[2] CLS. http://www.cs.unc.edu/˜maria/projects/CLS/.

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

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

[5] P. V. Bahl and V. 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, Mar. 2000.

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

[7] F. Chinchilla, M. Lindsey, and M. Papadopouli. Analysis of wireless information locality and association patterns in a campus. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Hong Kong, Mar. 2004.

[8] R. Harle and A. Hopper. Using personnel movements for indoor autonomous environment discovery. In *1st IEEE Conference on Pervasive Computing and Communications (PerCom)*, Fort Worth, TX, Mar. 2003.

[9] R. Harle, A. Ward, and A. Hopper. Single reflection spatial voting. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services*, San Francisco, May 2003.

[10] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, San Diego, Sept. 2003.

[11] J. Hightower and G. Borriello. A Survey and Taxonomy of Location Sensing Systems for Ubiquitous Computing. Technical report.

[12] J. Hightower, R. Want, and G. Borriello. SpotON: An indoor 3D location sensing technology based on RF signal strength. UW CSE tech report 2000-02-02, University of Washington, Seattle, WA, Feb. 2000.

[13] 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.

[14] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS). In *Proceedings of the IEEE Conference on Global Communications (GLOBECOM)*, San Antonio, Nov. 2001.

[15] D. Niculescu and B. Nath. Ad Hoc Positioning System (APS) using AoA. In *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, San Francisco,CA, Apr. 2003.

[16] M. Papadopouli and H. Schulzrinne. Effects of power conservation, wireless coverage and cooperation on data dissemination among mobile devices. In *ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc)*, Long Beach, California, Oct. 2001.

[17] N. B. Priyantha, A. K. L. Miu, H. Balakrishnan, and S. 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.

[18] C. Savarese, J. Rabaey, and K. Langendoen. Robust positioning algorithms for distributed ad-hoc wirleess sensor networks. In *Proc. of Usenix Annual Technical Conference*, June 2002.

[19] A. Savvides, W. Garber, S. Adlakha, R. Moses, and M. B. Srivastava. On the error characteristics of multihop node localization in ad-hoc sensor networks. In *2nd International Workshop on Information Processing in Sensor Networks (IPSN)*.

[20] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 166–179, Rome, Italy, July 2001.