Empirical-Based Analysis of a Cooperative Location-Sensing System

Konstantinos Vandikas Lito Kriara Tonia Papakonstantinou
Anastasia Katranidou Haris Baltzakis Maria Papadopouli *

Institute of Computer Science
Foundation for Research and Technology-Hellas

Department of Computer Science
University of Crete

ABSTRACT

We have designed a novel positioning system, the Cooperative Location-sensing system (CLS) that employs the peer-to-peer paradigm and a probabilistic framework to estimate the position of wireless-enabled devices in an iterative manner without the need for an extensive infrastructure or time-strenuous training. CLS can incorporate signal-strength maps of the environment to improve the position estimates. Such maps have been built using measurements that were acquired from Access Points (APs) and peers during a training phase. This paper makes three important contributions. First, it uses a particle-filters-based framework to model theoretically CLS. Second, it proposes new algorithms that incorporate real-life signal strength measurements from (APs) and peers to estimate position and distance. Third, it evaluates the performance of CLS via real-life measurements and extensive simulation, and compares it with other positioning systems. We have implemented and evaluated the CLS prototype along with its variants using IEEE802.11 and Bluetooth, and compared its performance with other positioning systems.

1. INTRODUCTION

Location-sensing has been impelled by the emergence of location-based services in the transportation industry, emergency situations for disaster relief, entertainment industry and assistive technology in the medical community. To support location-dependent services, a device needs to estimate its position. For example, the GPS-enabled navigation systems allow users to compute a route to guide them. However, GPS typically breaks down near obstacles, such as trees and buildings, and does not work indoors.

Location-sensing systems can be classified according to their dependency on, and use of, an infrastructure, specialized hardware, signal modalities, training, methodology and models for estimating distances, orientation, and position, coordination system, location description, localized or remote computation, scale, device identification, classification, recognition, cost, privacy, and accuracy and precision requirements [21]. The distance can be estimated using *time of arrival* (e.g., PinPoint [41]) or *signal strength* measurements, if the velocity of the signal and a signal attenuation model for the given environment, respectively, are known.

To infer the position, location-sensing systems devices may employ different modality, such as radio (Radar [8, 18], Ubisense[3], Ekahau [1]), infrared (Active Badge [36]), ultrasonic (Cricket [31, 32], Active Bat [33]), Bluetooth [19, 9, 16, 34, 10, 6, 18], 4G [35], or vision (EasyLiving [27, 2]), while others physical contact with pressure (Smart Floor), touch sensors or capacitive detectors.

A location-sensing system may infer the position using statistical analysis or pattern matching techniques on measurements acquired during training and run-time phase.

The wide popularity of the IEEE802.11 network, low deployment cost, and advantages of using it for both communication and positioning, make it an attractive choice. Most of the signal-strength based localization systems can be classified into the following two categories, namely the signature or map-based and the distance-prediction based. The first type creates a *signal-strength signature* or *map* of the physical space during a training phase and compares it with analogous run-time measurements [8, 28, 40]. To build such maps, signal strength data is gathered from beacons received from APs at various predefined checkpoints during a training phase. Thus, each checkpoint in the map associates the corresponding position of the physical space with statistical measurements based on signal strength values acquired at those positions. Such maps can be extended with data from different sources or signal modalities, such as ultrasound from deployed sensors to improve location-sensing [31, 19].

In other situations, a dense deployment of a wireless infrastructure for communication and location sensing may not be feasible due to environmental, cost, and regulatory barriers. Ad hoc networks exploit cooperation by enabling devices to

*Contact author (mgp@ics.forth.gr).

AUTONOMICS 2007, 28-30 October 2007, Rome, Italy

DOI 10.4108/ICST.AUTONOMICS2007.2226

share positioning estimates [38, 22, 11, 29, 12, 15, 41].

CLS is a novel location-sensing system using two features:

- the peer-to-peer paradigm
- probabilistic-based frameworks for transforming measurements from various sources to position and distance estimates

CLS applies the peer-to-peer paradigm by enabling devices to gather positioning information from other neighboring peers, estimate their distance from their peers based on signal strength measurements, and position themselves accordingly[17]. Periodically, CLS can refine its positioning estimations by incorporating newly received information from other devices.

CLS adopts a *grid-based* representation of the physical space; each cell of the grid corresponds to a physical position of the physical space. The cell size reflects the spatial granularity/scale. Each cell of the grid is associated with a value that indicates the likelihood that the node is in that cell. These values are computed iteratively using one of the following approaches:

- A simple *voting* algorithm, through which a local CLS instance casts *votes* on cells of the grid. A vote on a cell indicates the likelihood that the local device is located in the corresponding area of that cell.
- A particle filter-based model.

CLS can *incorporate additional information* to improve its location estimates. Examples of such information are: position estimates from different network interfaces (e.g., Bluetooth, RF tags, IEEE802.11), contextual semantics (e.g., topological information about the environment, mobility patterns, hotspots of the area), and signal-strength-based signatures of the physical space, to improve the location estimation.

This paper makes three important contributions. First, it employs a particle-filters-based framework to model CLS theoretically. Second, it proposes new algorithms that incorporate real-life IEEE802.11- and Bluetooth- signal strength measurements from wireless enabled devices to estimate the position. Third, it presents the CLS implementation and evaluates its performance via extensive simulations and real-life measurements.

Section 2 presents the communication protocol and section 3 the main CLS algorithms. Section 4 presents the particle filters based framework for CLS. In Section 5, we analyze the impact of the density of peers, and landmarks on the performance of CLS via simulations. Section 6 presents the prototype and main empirical-based performance results. Section 7 discusses related work and compares CLS with other location-sensing systems. Finally, Section 8 summarizes our main conclusions and plans for future work.

2. OVERVIEW OF CLS

CLS aims to enable devices to determine their location in a self-organizing manner without the need of an extensive infrastructure or training. The design of CLS was motivated by the following attractive properties, namely, tolerance to multiple network failures (e.g., AP failures or disconnections) and changes in the environment due to node mobility, ability to incorporate application-dependent semantics and various types of measurements, relative low computational complexity, use in both indoor and outdoor environments with pedestrian mobility, and scalability.

CLS can be integrated with a broad range of applications running on devices of different computing capabilities. Some of these devices may have *a priori* knowledge of their location that they can provide reliably. We refer to them as *landmarks*. A device that runs CLS to position itself is referred to as *node* or *non-landmark* peer.

3. VOTING ALGORITHM

A node tries to position itself on its local grid through a voting process in which devices participate by sending position information and casting votes on specific cells.

Each iteration of a local CLS instance (i.e., running at a peer) consists of the following steps:

Algorithm 1 An iteration of the voting process at a CLS instance

1. Gather position information from other peers
2. Record measurements from the received messages
3. Transform this information to a probability of being at a certain cell of its local grid
4. Add this probability to the existing value that this cell already has
5. Report a position that corresponds to the centroid of the set of cells with maximal weight

The process that transforms the acquired measurements into probability of being at a certain location can be implemented in different ways, described later on.

At the beginning of a run (Table 3), each peer broadcasts messages to *its one-hop neighbors* that include its *positioning information*, namely, its local id, maximum wireless range, and position, if known or computed. We refer to this broadcast update as *positioning message*.

We assume that an AP can act as a landmark and send positioning messages in the form of beacons. A peer records the signal strength values with which it receives these messages and responds by broadcasting its own position estimates.

Each local CLS instance employs an algorithm that transforms (maps) these signal strength values to either distance or position estimates. The transformation algorithm can be based on a radio attenuation model or a pattern matching algorithm. Such algorithms relate signal strength measurements acquired from messages exchanged between devices to their position on the terrain or their distance. Based on the position information of the sender and this distance estimation, the receiver estimates its own position on the local grid. When the local CLS estimates its own position,

it broadcasts this set of information, i.e., *CLS entry*, to its neighbors. Each node maintains a table with all the received CLS entries.

Figure 1: An example of the accumulation of votes on the grid cells of a host at different time steps. In each picture, the brighter an area, the more voting weight has been accumulated on the corresponding grid cells. The brightest area corresponds to a potential solution.

We denote the grid of the node k as G_k and as $v(i,j)$ the probability that the cell $(i,j) \in G_k$ is the position of node k. The region of the grid, $G_{h,k}$, i.e., set of cells for which peer k votes as possible region of node h.

Each node tries to position itself on its local grid. To determine its location, each node h gathers position estimations from other peers, and computes its own location using the following algorithm:

Algorithm 2 Position estimation at node h

1. Initialize the values of the grid G_h with all cells containing zeros.
2. If a signature of the environment is available, compare it with run-time measurements, and for each cell c of the grid, assign a vote of weight $w(c)$ (according to specified criteria).
3. For each received distance estimation at a peer k with a known or estimated position, perform the following steps:
   (a) Transform the coordinates of peer k to the coordinate system of the grid
   (b) Determine the region of the grid, $G_{h,k}$, i.e., set of cells for which peer k votes as possible region of node h. The determination can be based on a position-based or distance-based algorithm. If the peer k is a non-landmark, the distance between the two peers can be computed according to Eq. 2.
   (c) Increase the value of each cell in $G_{h,k}$ by v_k, where v_k is the voting weight of node k.
4. Assess the values of the cells in the grid and accept or reject the attempt for location sensing.

This is essentially a voting process, in which a node casts votes on the cells of its grid on behalf of other peers. Votes may have different *weights*. The larger voting weight a cell has acquired, the more likely it is for the local node to be located in the region that corresponds to that cell. The set of cells in the grid with maximal value indicates the *potential region*. 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.

When a training phase prior to voting is feasible, CLS can build a map or *signature* of a physical space, which is a grid-based structure of the space augmented with measurements from peers. This paper explores two such signature types, namely, *position-level* and *distance-level* signal-strength based signatures.

At run-time, the local CLS instance acquires signal-strength measurements from peers, constructs a run-time signature, and compares this run-time signature with the ones that have been generated during the training phase. We explore two different criteria for the comparison, namely, a *confidence interval*-based and a *percentiles*-based criteria.

3.1 Confidence interval-based criteria

During training, a position-level signature based on confidence intervals associates each position of the terrain (cell of the grid) with a vector of confidence intervals. Each entry of the vector corresponds to an AP and its associated confidence interval of the RSSI values that were recorded from beacons received from that AP during the training phase. Beacons are messages broadcast by APs periodically.

At run time, a local CLS instance acquires a number of beacons from APs and computes a confidence interval for each AP.

The algorithm assigns a weight at cell c $w(c)$ as,

$$w(c) = \sum_{i=1}^{n} (1 - \frac{(T_i^- - R_i^-) + (R_i^+ - T_i^+)}{R_i^+ - R_i^-}) \qquad (1)$$

where n is the total number of APs, R_i^+ and R_i^- and T_i^+ and T_i^- the upper and lower bound of the run-time and training confidence interval of AP_i, respectively.

Distance-based signatures are employed *only* when non-landmark peers participate in voting and training is possible. They relate distances with signal strength measurements recorded by the local CLS instance at the reception of messages from other non-landmark peers at the respective distances during training phase. During training, devices located at various positions participate in CLS by sending messages and recording the signal strength value with which each of these messages was captured and the distance between the two devices. Specifically, a training set is composed of entries, each including a distance and a confidence interval of the signal strength values recorded from messages exchanged between peers at that distance.

At run-time, the distance D between two peers is estimated

using the following formula:

$$D = \sum_{i=1}^{k} \frac{D_i * \sqrt{(R^- - T_i^-)^2 + (R^+ - T_i^+)^2}}{\sum_{i=1}^{k} \sqrt{(R^- - T_i^-)^2 + (R^+ - T_i^+)^2}} \quad (2)$$

where k is the number of entries in the training set, D_i is the i-th distance from the training set, $[R^-, R^+]$ is the runtime confidence interval and $[T_i^-, T_i^+]$ is the i-th confidence interval from the training set.

3.2 Percentiles-based criteria

Like the confidence interval, the percentile-based criteria uses also signal strength measurements. However, it captures more detailed information about their distribution, and thus, allows for more accurate comparisons. The weight of a cell c, $w(c)$, is computed as follows:

$$w(c) = \sum_{i=1}^{k} \sqrt{\sum_{j=1}^{p} (R_j - T_j^i(c))^2} \quad (3)$$

where k is the number of samples in the training set, p the number of percentiles, R_j the j-th real time percentile and $T_j^i(c)$ the j-th percentile from the i-th cell in the training set.

3.3 Discussion

Landmarks and nodes that are first to position themselves determine–to some extent–the accuracy of the location estimation of the remaining nodes, since their positioning estimates and errors are propagated in the network through the voting process. To minimize the impact of such errors, CLS imposes the following two conditions:

- The number of votes in each cell of the potential region must be above a threshold. We refer to this threshold as the *solution threshold* (ST).
- The number of cells in the potential region must be below a threshold, denoted as the *local error control threshold* (LECT).

In effect, ST controls how many nodes with known location must agree with the proposed solution. High ST reduces the error propagation throughout the network, but delays the positioning estimation. On the other hand, LECT determines the precision of each step. Another metric for filtering the local error can be the *diameter* of the region that corresponds to the maximum Euclidean of cells with the maximal value in voting weight.

Additional distance estimations from nodes with known location increase the voting weight and narrow down the potential region. The values for ST and LECT should be determined based on network characteristics, such as the *density of nodes*, and *landmarks*, and *accuracy of the distance estimations*. To prevent CLS from failing to report a position, both thresholds can be adaptively relaxed after rejecting potential solutions. Once the above conditions are satisfied, CLS reports the centroid of the potential region as the estimated location of the device.

CLS can be implemented in a centralized or distributed fashion, depending on whether or not the computations are performed on a server or peers. Furthermore, in the centralized case one or more servers can be deployed depending on the topography of the terrain.

4. PARTICLE FILTER-BASED FRAMEWORK

In probabilistic terms, CLS can be formulated as the problem of determining the probability of a node being at a certain location given a sequence of signal strengths. Assuming first-order Markov dynamics, the above problem can be expressed using the network graph depicted in Figure 2, where x_k is the node location (system state) at time instant $k = 1, \ldots, T$. x_k cannot be observed directly (it is "hidden"). Yet, for each location x_k, a measurement vector y_k (signal strength) is available that depends on the hidden variable according to a known observation function.

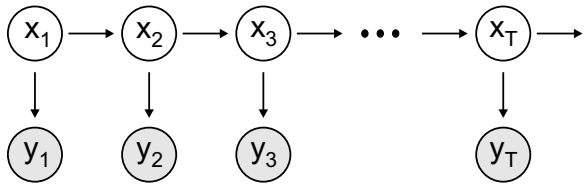

Figure 2: State space model for the proposed location sensing system. Clear circles indicate hidden state variables, grayed circles indicate observations, horizontal arrows indicate state transition functions and vertical arrows indicate observation functions.

Due to the Markov assumption, each node location, given its immediately previous location, is conditionally independent of all earlier locations, that is

$$P(x_k|x_0, x_1, \ldots, x_{k-1}) = P(x_k|x_{k-1}). \quad (4)$$

Similarly, the observation at the k-th time instant, given the current state, is conditionally independent of all other states

$$P(y_k|x_0, x_1, \ldots, x_k) = P(y_k|x_k). \quad (5)$$

Based on the this model, location-sensing can be formulated as the problem of computing the location x_k of a node at time k, given the sequence of observations $y_1, y_2, ...y_k$, up to time k, that is, determining the *a posteriori* distribution $P(x_k|y_1, y_2, \ldots, y_k)$.

To estimate the above *a posteriori*, which is actually a density over the whole state space, we use particle filter. Particle filter is a technique for implementing a recursive Bayesian filter by Monte Carlo sampling. According to this technique, the *a posteriori* $P(x_k|y_1, y_2, \ldots, y_k)$ is expressed as a set of samples

$$x^{(L)} = (x, y)^{(L)}, L = 1, 2, \ldots, N \quad (6)$$

distributed among the whole state-space. The denser the samples in a certain region of the state-space, the higher the probability that the node is located in that region.

Unlike Kalman-filter, particle filter does not impose any constraints on the format of the involved distributions and noise models, or on linearity of the involved functions. This makes them particularly well-suited to location-sensing.

4.1 Sampling/Importance Resampling (SIR) algorithm

To generate and maintain the samples (particles), we utilize the Sampling/Importance Resampling (SIR) algorithm introduced by Rubin [37]. According to SIR, instead of sampling the true *a posteriori* distribution (which is not possible because this distribution is not available in closed form), samples are drawn from the so-called proposal distribution $\pi(x_k|y_1, y_2, \ldots, y_k)$. To compensate for this difference, each sample $s^{(L)}$ is also assigned a weight $w^{(L)}$, which is computed, according to the Importance Sampling Principle:

$$w_t^{(L)} = \frac{P(x_k|y_1, y_2, \ldots, y_k)}{\pi(x_k|y_1, y_2, \ldots, y_k)} \tag{7}$$

By choosing the proposal distribution to be the transition prior $P(x_k|x_{k-1})$, the weights can be computed as

$$w_t^{(L)} = \frac{P(x_k|y_1,y_2,\ldots,y_k)}{\pi(x_k|y_1,y_2,\ldots,y_k)} \tag{8}$$

$$\approx \frac{P(y_k|x_k)P(x_k|x_{k-1})}{P(x_k|x_{k-1})} w_{t-1}^{(L)} \tag{9}$$

$$= P(y_k|x_k) w_{t-1}^{(L)} \tag{10}$$

To avoid degenerate situations in which large numbers of samples have weights close to zero, after a few iterations, SIR also includes a resampling step which ensures that unlikely samples are replaced with more likely ones. The following pseudocode describes our implementation of SIR.

Algorithm 3 SIR algorithm

1: For $L = 1, \ldots, P$
2: Transition step: Draw a new sample $x_k^{(L)}$ from the transition prior of sample $x_{k-1}^{(L)}$ according to $P(x_k^{(L)}|x_{k-1}^{(L)})$
3: Observation step: Calculate the weight $w_k^{(L)}$ of sample $x_k^{(L)}$, according to the importance sampling principle. That is, $w_k^{(L)} = w_{k-1}^{(L)} \cdot P(y_k|x_k^{(L)})$
4: End for loop
5: Normalize weights
6: Resample
7: Goto Step 1

Initially, all particles are uniformly distributed among the state space. Particles whose their validity is confirmed by observations tend to get larger weights, so after a few resampling steps, particles are expected to gather around certain regions. If sufficient input is available to resolve all ambiguities, all particles will eventually gather in one region only.

The density of particles in a specific region of the state-space indicates the probability of the node to be in that region. The expected node location is defined as the location within the state-space with the highest particle density and is computed iteratively using the *highest particle density* algorithm. In this algorithm, the particle with the highest number of other particles within a certain radius is initially chosen as the circle center. The centroid of these supporting particles is then calculated and the process is repeated with this calculated centroid being the circle center until convergence is achieved.

4.2 Computation of transition prior

Motivated by the observation that nodes located in offices tend to remain mostly stationary while nodes in corridors tend to be in motion, we segment the state space into two different types of area: corridor and office areas.

To compute the transition prior $P(x_k^{(L)}|x_{k-1}^{(L)})$ for the Step 2 of SIR, we consider the following cases:

1. the particle's current position estimate $x_k^{(L)}$ being inside an office area
2. the particle's current position estimate $x_k^{(L)}$ being in a corridor area

The transition prior for office areas is assumed to be a gaussian distribution, centered at the previous position estimate $x_{k-1}^{(L)}$ and having a standard deviation of σ_{off}. For corridor areas, the transition prior was assumed to be uniform within a radius δ_{cor} around the previous position estimate defined as follows:

$$P(x_k^{(L)}|x_{k-1}^{(L)}) = \begin{cases} \frac{1}{\pi\delta_{cor}^2}, & if ||x_k^{(L)} - x_{k-1}^{(L)}|| \leq \delta_{cor} \\ 0, & otherwise \end{cases} \tag{11}$$

The standard deviation σ_{off} of the "office" transition prior and the radius δ_{cor} for the "corridor" transition prior were found experimentally and were set to be 0.5m and 1m, respectively.

4.3 Computation of observation probability

To compute the observation probability $P(y_k|x_k^{(L)})$, required for Step 3 of the SIR algorithm, we utilize observations from both APs and other peers.

For APs, observation models were extracted and stored offline during training, as described in Section 3. That is, the state-space is discretized into a finite number of cells. For each state-cell, the mean signal strength of each AP and its corresponding standard deviation is computed and stored. For cells with no measurements, we apply interpolation and store the reported values.

For peers, signal strength values measured during training are stored as a function of distance. As in the case of APs, the distance-space is discretized, and for each distance, the mean and standard deviation of the observed signal strengths are stored. It is important that the gathered training data captures all the variation in different positions within the state space.

Although peer training data exhibits large variation, it can improve the accuracy by considering that the distance of two peers is likely to exceed a certain threshold, when the corresponding signal strength measurements drop to zero.

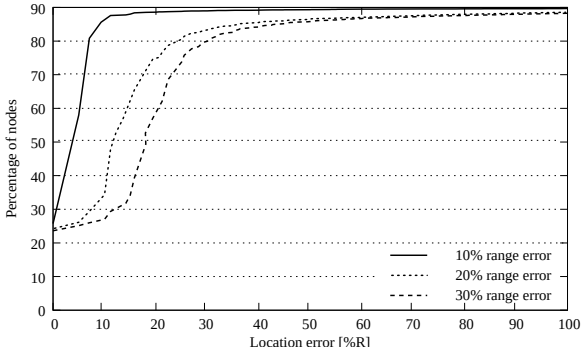

Figure 3: Accuracy level as a function of the range error. 10% of peers are landmarks.

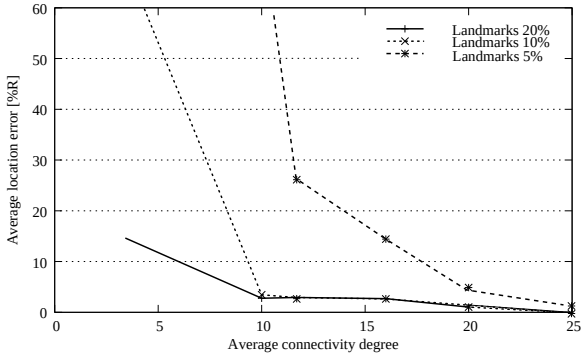

Figure 4: Sensitivity study of the impact of various connectivity degrees and landmark percentage. The terrain includes 100 peers with a range error of 5%.

Based on the assumption that signal strength measurements are independent, at run-time, all available signal strength measurements are combined to compute the joint observation probability as:

$$p(y_k|x_k) = p((y_{k1}, y_{k2}, \ldots, y_{kn})|x_k) \quad (12)$$

$$= p(y_{k1}|x_k) \cdot p(y_{k2}|x_k) \cdot \ldots \cdot p(y_{kn}|x_k), \quad (13)$$

where n is the number of measurements.

5. SIMULATION-BASED EVALUATION

For the evaluation of CLS, we investigated the impact of the range error in distance estimation, connectivity degree, and percentage of landmarks, grid size, and mobility on the accuracy level through extensive simulations using the *ns-2*.

The simulated network included N peers that had been distributed independently of each other with coordinates selected according to a uniform distribution using a pseudo-random generator in a grid of size AxA cells. Two peers can communicate if they are within a distance of r. The average degree of connectivity is λ (see Appendix).

We set the voting weight of the landmarks equal to 100 and of the remaining peers equal to one. Both the location and range error are expressed in terms of the transmission range. For example, 50% location error means an error equal to half of the transmission range of the wireless infrastructure.

For the simulations, we assumed that CLS can estimate the distance between two peers with a given *range error* that remains fixed in each simulation scenario. To evaluate its impact, we varied its value to be equal to 10%, 20% and 30% of the transmission range. Unless otherwise stated, a range error of 10% is assumed. Finally, for each different scenario, 100 simulation runs were performed and their average value was reported.

In the case of a 10% range error, 85% of peers have 10% at most location error. 80% of peers have 20% (30%) at most location error, for range error equal to 20% (30%) range error, respectively (Figure 3). For a high range error equal to the half of the transmission range, the average position error is 25% of the transmission range, and for a range error equal to 10%, 20% and 30%, the average location error is 5%, 10% and 18%, respectively.

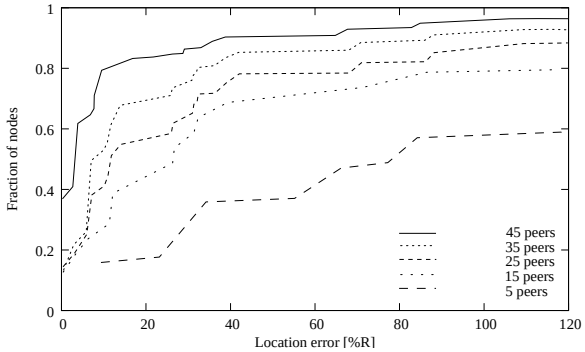

Figure 5: Location error for various degrees of connectivity. In all the scenarios, there are five APs and the range error is 10%.

To investigate the impact of the degree of connectivity and percentage of landmarks, we fixed the total number of peers and range error, and varied the transmission range and percentage of landmarks. The lower the connectivity degree and fewer the landmarks, the higher the location error. However, in the case of 10% landmarks or more, and connectivity degree of at least 7, the location error is reduced considerably. As illustrated in Figure 4, landmarks have a strong impact on the accuracy level of CLS. More particularly, when the number of landmarks is 10% or higher, the location error decreases dramatically. In addition, the average connectivity degree has a higher impact on the average location error when the percentage of landmarks is less than a threshold of 10%. For instance, in the case of 10% landmarks, a 50% increase in the average connectivity degree corresponds to a decrease of a location error higher than 50%, whereas, for a 5% landmarks, the decrease is dramatic.

The smaller the cell size in the grid, the higher the accuracy but also the memory and CPU power requirements. We found that a grid resolution of 100x100 cells balances the computational complexity and accuracy tradeoff.

6. EMPIRICAL-BASED EVALUATION

6.1 Testbed description

The training and run-time experiments took place in the Telecommunication and Networks Lab (TNL) at the Foun-

dation for Research and Technology-Hellas (FORTH) in Greece, an area of 7m × 12m. For this area, a grid-based structured was considered with cells of size 50cm×50cm.

To generate the signal strength signatures for training, CLS acquiers 30 samples of signal strength values for each cell of the grid in which signatures were generated (shown in Figure 6). In these cells, the trainer waited for approximately one minute to acquire the training signal strength measurements. There were 11 APs in total, out of which 3.5 APs, on average, can be detected at a given cell.

To capture signal strength values, the following monitoring tools were used:

1. IWLIST [5], which polls each channel and acquires the mac address and signal strength measurements from each AP (in dBm).
2. TCPDUMP[4], a passive scanner relying on libpcap [4] for the retrieval of each packet.

The experiments were conducted around 3pm on several weekdays throughout the semester, during which there were at least five people in the laboratory, and several others walking in the hallways outside. A Sony Vaio and Fujitsu Siemens Tablet PC with the same wireless adapter were used for both the training and run-time experiments. Figure 6 shows a map of the TNL, where the CLS implementations and Ekahau were tested.

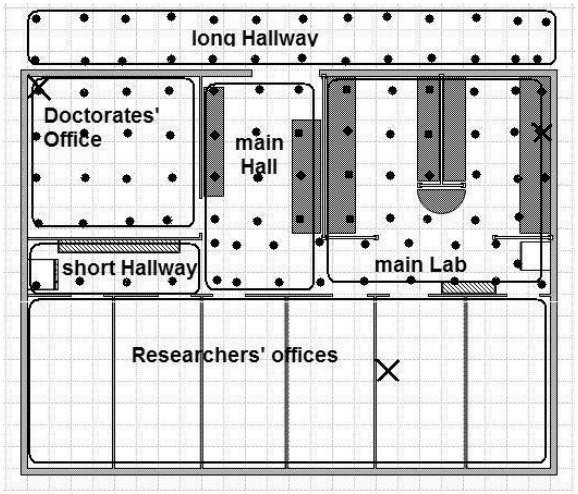

Figure 6: TNL floorplan. Dots indicate the positions at which measurements in the training phase were acquired and crosses the Bluetooth hotspots.

CLS was implemented in C++ and evaluated with real-life measurements. Specifically, the following CLS variations were developed: Like CLS, Ekahau also uses the IEEE802.11 infrastructure, creates a map with calibration data, and compares the training and run-time measurements to estimate the position.

We found that CLS-10 has a median location error of 1.8 m while for 90% of the experiments, the location error is at most 3.5 m (Figure 7). For 90% of the tested positions, the location error of Ekahau v3.1 is at most 4.6 m, with a median location error of 2.3 m. As expected, CLS-1 has a higher location error than CLS-10, trading accuracy with response time. Specifically, the median location error of CLS-10 and CLS-1 are 1.8 m and 2.4 m, respectively.

CLS-percentiles uses more detailed information about the signal strength, and thus, performs better than CLS-1. The experiments confirm that observation, indicating a median location error of 2m, with a 95% confidence interval of [1.88, 2.44] (for the experiment 1). We repeated the CLS experiment one week later, during a different time of day (experiment 2). As shown in Figure 9, there is no significant performance difference of CLS-percentile between these two experiments.

To evaluate the impact of the number of peers on the per-

CLS-1 which uses the confidence interval based criteria, in which only APs vote to estimate the position of the local device. It performs *one iteration* and reports the position estimate.
CLS-10 which uses the confidence interval based criteria in which only APs vote to estimate the position of the local device. It performs *ten consecutive iterations*, and reports the mode of the estimated positions.
CLS-p2p which includes an *additional* em non-landmark peer in experiments and employs distance-level measurements to estimate its distance from each other. Thus, apart from APs, there are two peers participating in the evaluation.
CLS-particles which uses the particle filter-based model. *Only* APs vote.
CLS-percentiles which has only one difference from CLS-1: CLS-percentiles uses percentiles of signal strength values instead of confidence interval-based criteria in voting.

Table 1: CLS variations.

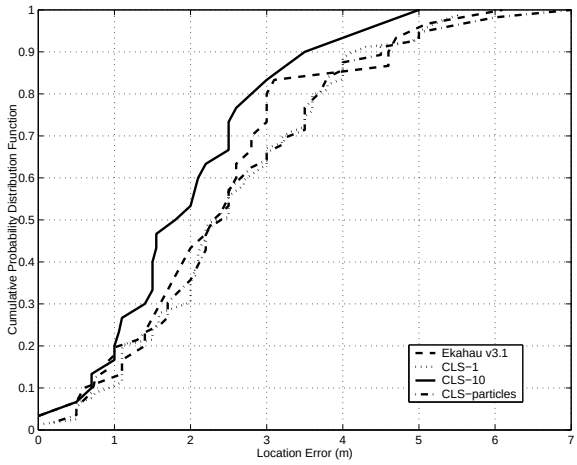

Figure 7: Location error of CLS-10, CLS-1, CLS-particles and Ekahau v3.1 in the TNL.

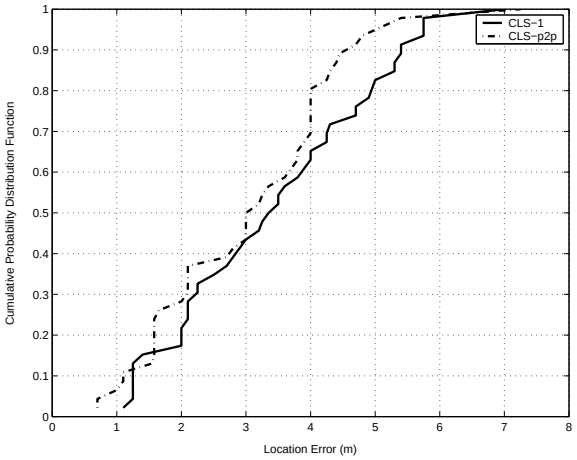

Figure 8: Impact of one peer on the accuracy of location estimation.

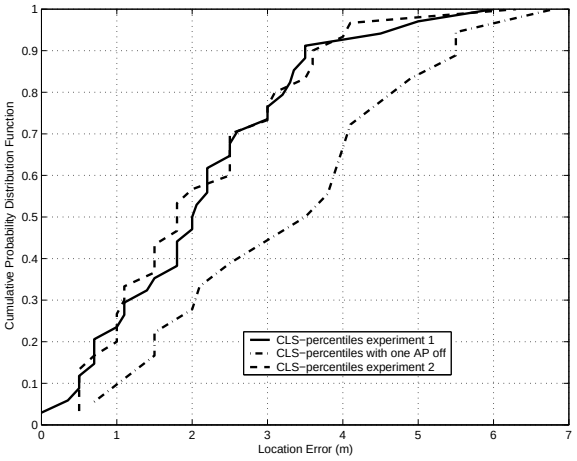

Figure 9: Difference in the location error of CLS-percentile when one AP is absent at run time.

formance, we ran CLS experiments varying the number of participating peers (at run-time). Specifically, in one set of experiments, there was one AP absent in the run-time measurements. In this case, the median location error is 3.5 m compared to 2 m when the AP is present.

In a different scenario, a non-landmark peer is added and new experiments were performed with a total of 11 APs and two non-landmark peers running the CLS-p2p. The two non-landmark peers perform the distance-based signatures. As shown in Figure 8, this improves the CLS performance; 80% of the experiments reported an error up to 4 m compared to 5 m for CLS-1.

7. RELATED WORK

Recently, significant work has been published in the area of location-sensing using RF signals. Like CLS, Radar [8] employs signal strength maps that integrate signal strength measurements acquired during the training phase from APs at different positions with the physical coordinates of each position. Each measured signal strength vector is compared against the reference map and the coordinates of the best match will be reported as the estimated position. 90% of the time the hosts can be located with at most six meters of error with a sampling density of one sample every $13.9\,m^2$ and $19.1\,m^2$ of their testbed using the signal from three and five APs, respectively.

Bahl *et al.* extended Radar to alleviate side effects that are inherent properties of the signal strength nature, such as aliasing and multipath [7]. The extended Radar testbed with the Cisco Aironet 4800 series had a mean of 2.37 m and a 90% percentile of 5.97 m.

Ladd *et al.* [28] proposed another location-sensing algorithm that utilizes the IEEE802.11 infrastructure. In its first step, a host employs 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 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 1.5 m.

Niculescu and Badri Nath [30] designed and evaluated a cooperative location-sensing system that uses specialized hardware for calculating the angle between two hosts in an ad hoc network. 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. Previously, these authors [29] introduced a cooperative location-sensing system in which 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 evaluated three variations, namely the "DV-hop", "DV-distance", and "Euclidian". Though CLS 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 "Euclidian" scheme, but with always 100% solved hosts. CLS differs from this work mainly in the grid-based representation of the environment and voting algorithm.

Another location-sensing system in ad hoc networks performs positioning without the use of landmarks or GPS and presents the tradeoffs among internal parameters of the system [11]. The location-sensing systems presented in [38] and [20] are the closest to CLS and are compared in detail in [17].

Active Badge uses diffuse infrared technology, and requires each person to wear a small infrared badge that emits a globally unique identifier every ten seconds or on demand. A central server collects this data from fixed infrared sensors around the building, aggregates it and provides an application programming interface for using the data. The system suffers in the case of fluorescent lightning and direct sunlight, because of the spurious infrared emissions these light sources generate.

SmartFloor employs a pressure sensor grid installed in all

floors to determine presence information. It can accurately determine positions in a building without requiring from users to wear tags or carry devices, but also not able to specifically identify individuals. UbiSense provides an accuracy of 15 cm using a network of ultra wide band (UWB) sensors (17 cm×12 cm×5 cm) installed and connected into a building's existing network (four sensors in a typical office environment of 625 m^2). The UWB sensors use Ethernet for timing and synchronization. They detect and react to the position of tags based on time difference of arrival and angle of arrival. An RF tag is a silicon chip that emits an electronic signal in the presence of the energy field created by a *reader* device in proximity. Location can be deduced by considering the last reader to see the card. RFID proximity cards are in widespread use, especially in access control systems.

The Active Bats system consists of a controller that sends a radio signal and a synchronized reset signal simultaneously to the ceiling sensors using a wired serial network. Bats respond to the radio request with an ultrasonic beacon. Ceiling sensors measure time-of-flight from reset to ultrasonic pulse. ActiveBat applies statistical pruning to eliminate erroneous sensor measurements caused by a sensor hearing a reflected pulse instead of one that travelled along the direct path from the Bat to the sensor. A relatiVely dense deployment of ultrasound sensors in the ceiling can provide within 9 cm of true position 95% of the measurements.

Zaruba *et al.* [39] performed localization based on the Received Signal Strength Indication (RSSI) of signal measurements gathered from two APs at various predefined locations and orientations of an indoor environment. Based on them, they built a grid-based signal strength map and used the SIR algorithm of particle filters. In particularly, they deployed 3000 particles in an area of 88 cells, and reported a mean location error of at most 2.1 m. In their real-time tests, the mean location error was at most 6 m using 5000 particles. Their training phase has a large overhead, because it takes place at each cell of the map and for different orientations.

Evennou and Marx [13] employed Kalman and particle filters with the Motley-Keenan propagation model. They built a signal strength map acquiring one measurement in each room and one measurement in every two meters in a corridor of a 35x35 m^2 area with four APs placed at each corner. Kalman and particle filters reported a mean positioning accuracy of 2.29 m and 1.86 m, respectively. However, sometimes after a re-sampling step all the particles were trapped in a part of the building and always bumped into a wall after then. In [14], they incorporated additional information from Inertial Navigation Systems (INS), improving the mean location error to 1.53 m using 10,000 particles.

Hightower and Borriello [23] also applied particle filters that use the Sequential Importance Sample with Resampling (SISR) algorithm for indoor positioning. The appropriate number of samples is determined at each step using a procedure called KLD adaptation. The authors used a robot that walks with a speed in the range of [0,2] m/s and collects measurements from a WiFi client device, an ultrasound badge, two types of infrared badges, RFID tags and then tested the system in a 900 m^2 office building. The estimated position was computed as the weighted mean of its samples. The 80% location error was at most 1.8 m. They experimented with particle filter algorithms running on devices ranging from high-end servers to PDAs.

In robotics, localization has been a research topic of significant interest. For example, Howard *et al.* [24], considered IEEE802.11-enabled mobile robots armed only with a signal strength map and odometry, that can localize themselves using a variant of the standard Monte Carlo Localization algorithm. They encoded the signal strength map using a regular grid and compared the results achieved using IEEE802.11, contact, and a combination of them. Letchner *et al.* [26] introduced a hierarchical Bayesian technique for learning local Gaussian likelihood models of signal strength and integrated their model into a graph-based location estimation system. They experimented in IEEE802.11 indoor and outdoor environments.

In [18], Gwon *et al.* proposed two indoor location-estimation algorithms based on RF technology that merge information from other location-estimation techniques to improve location accuracy. Their first algorithm combines multiple information sources to find the location of stationary users while the second addresses the problem of aliasing when two locations have similar RF characteristics. They evaluated their algorithms in a corporate environment that includes cubicles and small offices using four IEEE802.11 APs, three Bluetooth APs and a laptop. Their algorithms improved the location accuracy by 24% or more.

Ekahau Positioning Engine (EPE) [1] is primarily used for indoor tracking, but could be also used within outdoor applications. Like CLS, Ekahau also uses the IEEE802.11 wireless infrastructure, creates a map, records site calibration data, and compares the training and run-time measurements to estimate the position.

Hu and Evans adapted the Sequential Monte Carlo localization and particle filters in a setting with landmarks and hosts that estimate their location from messages they receive[25].

Like CLS, Horus WLAN [40] operates in two phases: the training phase and the online phase. The main difference between the two systems lies in the number of samples they require in each phase for the creation of the training map. As Horus WLAN attempts to find the correlation between consecutive samples using an autoregressive model, a high number of samples are required for the creation of a valid model. However, the authors do not provide extensive results on the number of samples or the impact of the sample size to the location estimation. Furthermore, a high number of samples stresses the amount of time it takes for the system to be trained and also to estimate the user's location. CLS requires only 30 samples for each cell of the training set (Figure 6), during the training phase, and 10 or 30 samples in runtime depending on the location estimation algorithm. The trainer needs to wait for approximately 1 minute in each cell during the training phase. In a 11.8 m by 35.9m area with a total coverage of 5 APs and an average of 4 APs per cell, Horus WLAN has a 90th percentile of 1.32 m estimation error. In a similar area, with a total of 11 APs and an

average of 3.5 APs per cell CLS has a 90th percentile of 3m estimation error and 6 to 15 seconds to reach a conclusion on the user's location.

8. CONCLUSIONS AND FUTURE WORK

CLS uses the peer-to-peer paradigm and a probabilistic-based framework to position wireless-enabled devices in a self-organizing manner. These features allow CLS to easily incorporate information from various sources, such as contextual information, measurements from heterogeneous devices (e.g., Bluetooth and IEEE802.11 wireless interfaces), positioning information from other peers, signal-strength maps of the environment, to enhance its positioning accuracy. The probabilistic-based framework uses a grid-like representation of the environment and a voting algorithm.

Several CLS variants have been implemented and evaluated via extensive simulations and empirical-based measurements. For the empirical-based evaluation, we run experiments using IEEE802.11 signal strength measurements. CLS has a satisfactory accuracy level without the need of specialized hardware and extensive training. It can be easily extended for outdoor environments and different mobility patterns.

We found that the density of landmarks and peers has a dominant impact on positioning. CLS can utilize signal strength maps of the physical space by superimposing statistical properties of the signal strength values acquired during the training phase on their corresponding positions. Such maps can significantly improve its performance. Through empirical-based experiments, we showed how the different statistical analysis properties of signal-strength, the particle-filters model, the AP failures and additional peers affect the performance of CLS. Currently, the training is static, in that it does not consider the placement of rogue or new APs and changes in the configuration, position or orientation of APs and density of users or objects in the area. Such changes may affect the signal strength and the signal strength matching process. One of our objectives is to enable CLS to perform a calibration phase dynamically to detect some of these changes and incorporates them in the map. The tradeoff between the increased complexity and overhead of the training and runtime phases and the improvements in the accuracy and precision needs to be addressed.

Also, it would be interesting to use criteria based on chi-square tests and the Kullback-Leiber divergence in the voting process and evaluate their performance. Chi-square tests will allow us to compare the training and run-time signal strength distributions in more detail.

Our simulation results indicate that topological information about the environment (e.g., about hotspot areas, presence information of users, existence of walls, user mobility patterns) can have a dramatic impact in the performance of the system. Part of our future research effort is to incorporate such heuristics in the probabilistic frameworks of CLS and extend the performance analysis study.

9. APPENDIX

A *geometric random graph*, $G(N, r)$ can model such networks. By holding λ to be a constant equal to $(N-1)\pi r^2/A^2$, the probability of a node having degree k can be calculated as $p(k)$, where

$$p(k) \approx e^{-\lambda}(\lambda)^k/k!, \quad (14)$$

with the last approximate equality becoming exact in the limit of large A and N, and fixed k. The average degree of connectivity is λ.

10. ACKNOWLEDGMENT

This work was supported by the Greek General Secretariat for Research and Technology (05NON-EU-238 and CRETE-WISE KP-18), and the European Commission (MIRG-CT-2005-029186). We would also like to thank Konstantinos Mastorakis for his contribution to the extension of CLS using the Bluetooth technology.

11. REFERENCES

[1] Ekahau v.3.1. (http://www.ekahau.com).
[2] Microsoft EasyLiving. http://research.microsoft.com/easyliving/.
[3] Precise Real-time Location. http://www.ubisense.net/.
[4] tcpdump/libpcap. (http://www.tcpdump.org/).
[5] Wireless tools for linux. (http://www.hpl.hp.com/personal/).
[6] S. Asthana and D. Kalofonos. The problem of bluetooth pollution and accelerating connectivity in bluetooth ad-hoc networks. In *Proc. of IEEE International Conference on Pervasive Computing and Communications (Percom)*, New York, NY, Mar. 2005.
[7] P. Bahl, V. N. Padmanabhan, and A. Balachandran. Enhancements to the radar user location and tracking system, Feb. 2000.
[8] P. V. Bahl and V. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *IEEE Conference on Computer Communications (Infocom)*, Tel Aviv, Israel, Mar. 2000.
[9] U. Bandara, M. Hasegawa, M. Inoue, H. Morikawa, and T. Aoyama. Design and implementation of a bluetooth signal strength based location sensing system. In *IEEE Radio and Wireless Conference (RAWCON)*, Atlanta, GA, Sept. 2004.
[10] R. Bruno and F. Delmastro. Design and analysis of a bluetooth-based indoor localization system. Technical report, Pisa,Italy, 1999.
[11] S. Capkun, M. Hamdi, and J. P. Hubaux. GPS-Free Positioning in Mobile Ad-Hoc Networks. In *Proceedings of HICSS*, Hawaii, Jan. 2001.
[12] K. Chintalapudi, R. Govindan, G. Sukhatme, and A. Dhariwal. Ad-hoc localization using ranging and sectoring. In *IEEE Conference on Computer Communications (Infocom)*, March 2004.
[13] F. Evennou and F. Marx. Improving positioning capabilities for indoor environments with WiFi. In *EUSIPCO 2005*, Antalya, Turkey, Sept. 2005. IST.
[14] F. Evennou and F. Marx. Advanced integration of wifi and inertial navigation systems for indoor mobile positioning, 2006. EURASIP Journal on Applied Signal Processing.
[15] L. Fang, W. Du, and P. Ning. A beacon-less location discovery scheme for wireless sensor networks. In *IEEE Conference on Computer Communications (Infocom)*, pages 161–171, Miami, Florida, Mar. 2005.
[16] S. Feldmann, K. Kyamakya, A. Zapater, and Z. Lue. An indoor Bluetooth-based positioning system: concept, implementation and experimental evaluation. Technical report, University of Hannover, 2003.
[17] C. Fretzagias and M. Papadopouli. Cooperative Location Sensing for Wireless Networks. In *Second IEEE International conference on Pervasive Computing and Communications*, Orlando, Florida, Mar. 2004.
[18] Y. Gwon, R. Jain, and T. Kawahara. Robust indoor location estimation of stationary and mobile users. In *IEEE Conference on Computer Communications (Infocom)*, Mar. 2004.
[19] Y. Gwon, R. Jain, and T. Kawahara. Robust indoor location estimation of stationary and mobile users. In *IEEE Conference on Computer Communications (Infocom)*, March 2004.
[20] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale

sensor networks. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, San Diego, Sept. 2003.

[21] J. Hightower and G. Borriello. A Survey and Taxonomy of Location Sensing Systems for Ubiquitous Computing. Technical Report, University of Washington, Department of Computer Science and Engineering UW CSE 01-08-03, Seattle, WA, 2001.

[22] 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.

[23] Hightower, J. and Borriello, G. Particle Filters for Location Estimation in Ubiquitous Computing: A C ase Study. In *Proceedings of the Sixth International Conference on Ubiquitous Computing (Ubicomp '04)*, Nottingham, England, September 2004.

[24] A. Howard, S. Siddiqi, and G. Sukhatme. An experimental study of localization using wireless ethernet. In *International Conference on Field and Service Robotics*, Yamanaka, Japan, July 2003.

[25] L. Hu and D. Evans. Localization for mobile sensor networks. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[26] Julie Letchner and Dieter Fox and Anthony LaMarca. Large-Scale Localization from Wireless Signal Strength. In *AAAI*, Pittsburgh, Pennsylvania, USA, July 2005.

[27] J. Krumm, S. Harris, B. Meyers, B. Brumitt, M. Hale, and S. Shafer. Multi-camera multi-person tracking for easyliving. In *Proceedings of the Third IEEE International Workshop on Visual Surveillance*, Washington, DC, USA, 2000. IEEE Computer Society.

[28] A. M. Ladd, K. Bekris, A. Rudys, G. Marceau, L. E. Kavraki, and D. Wallach. Robotics-based location sensing using wireless ethernet. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Atlanta, GE, Sept. 2002. ACM Press.

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

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

[31] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. Boston, MA, Aug. 2000.

[32] N. B. Priyantha, A. K. L. Miu, H. Balakrishnan, and S. Teller. The cricket compass for context-aware mobile applications. In *ACM International Conference on Mobile Computing and Networking (MobiCom)*, pages 1–14, Rome, Italy, July 2001.

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

[34] M. Rodriguez, J. P. Pece, and C. J. Escudero. In-building location using bluetooth. In *International Workshop on Wireless Ad-hoc Networks 2005*, Coruna,Spain, May 2005.

[35] A. Roy, A. Misra, and S. K. Das. An information-theoretic framework for optimal location tracking in multi-system 4G wireless networks. In *IEEE Conference on Computer Communications (Infocom)*, Hong Kong, Mar. 2004.

[36] Roy Want and Andy Hopper and Veronica Falcao and Jon Gibbons. The Active Badge Location System. 1992. ACM Transactions on Information Systems.

[37] Rubin, D. B. Using the SIR algorithm to simulate posterior distributions. in Bayesian Statistics 3, eds. J.M. Bernardo, M.H. DeGroot, D.V. Lindley, and A.F.M. Smith, Cambridge, MA: Oxford University Press, pp. 395–402.

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

[39] V. Seshadri, G. V. Zaruba, and M. Huber. A bayesian sampling approach to in-door localization of wireless devices using received signal strength indication. In *Proc. of IEEE International Conference on Pervasive Computing and Communications (Percom)*, Kauai Island, Hawaii, Mar. 2005.

[40] M. Youssef and A. Agrawala. The horus wlan location determination system. In *International onference on Mobile Systems, Applications and Services (MobiSys)*, Seattle, USA, 2005.

[41] M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala. PinPoint: An asynchronous time-based location determination system. In *International onference on Mobile Systems, Applications and Services (MobiSys)*, pages 165–176, 2006.