Adaptive Lookup of Open WiFi Using Crowdsensing

Di Wu, *Member, IEEE, ACM*, Qiang Liu, Yong Li, *Member, IEEE, ACM*, Julie A. McCann, *Member, IEEE, ACM*, Amelia C. Regan, *Member, IEEE*, and Nalini Venkatasubramanian, *Senior Member, IEEE, ACM*

***Abstract*— Open WiFi access points (APs) are demonstrating that they can provide opportunistic data services to moving vehicles. We present `CrowdWiFi`, a novel system to look up roadside WiFi APs located outdoors or inside buildings. `CrowdWiFi` consists of two components: online compressive sensing (CS) and offline crowdsourcing. Online CS presents an efficient framework for the coarse-grained estimation of nearby APs along the driving route, where received signal strength (RSS) values are recorded at runtime, and the number and location of the APs are recovered immediately based on limited RSS readings and adaptive CS operations. Offline crowdsourcing assigns the online CS tasks to crowd-vehicles and aggregates answers on a bipartite graphical model. Crowd-server also iteratively infers the reliability of each crowd-vehicle from the aggregated sensing results, and then refines the estimation of the APs using weighted centroid processing. Extensive simulation results and real testbed experiments confirm that `CrowdWiFi` can successfully reduce the computation cost and energy consumption of roadside WiFi lookup, while maintaining satisfactory localization accuracy.**

***Index Terms*— Localization, crowdsensing, vehicular networks.**

I. Introduction

ROADSIDE WiFi networks are increasingly being tapped into by end users with WiFi interfaces in vehicular networks opportunistically for a broad range of applications including ad hoc data dissemination and low-cost Internet access [1], [2]. These networks use fixed access points (APs) that provide improved higher bandwidth connectivity due to better signal propagation characteristics and their ability to exploit spare spectrum. This is especially the case in locations with limited cellular coverage and/or in environments vulnerable to the obstruction of satellite signals by buildings and is typical in both urban environments (with significant built infrastructure) and in rural areas (where cellular connectivity may be sparse) [3], [4].

With the growing popularity of open WiFi community, people could turn their home or business WiFi APs into open WiFi APs on their own by installing third-party firmware such as OpenWrt [5] or DD-WRT [6], and then create a private WiFi sale store for surrounding users to access the guest network. The alternative to set up open WiFi is to share wireless connections via an intermediary company such as Fon [7] or BT [8]. The company's members agree to share a part of their WiFi bandwidth, so that they can connect to other members' open WiFi. For consumers who choose not to share their WiFi connection, they can buy WiFi access passes or credit from the company. The company's members whose open WiFi APs are used by the paying customers can receive part of the revenue. Fon claims to have the largest open WiFi network in the world, with over 17 million APs as at August 2015. Given such a successful development of open WiFi community, it is possible for a moving vehicle to have shared Internet connection via roadside WiFi networks with the assistance of on-board mobile systems, such as recently launched GM OnStar [9] services in commercial vehicles. In comparison with dedicated short-range communication (DSRC) as defined in IEEE 1609 for safety-related messaging between infrastructure and vehicles in a secure manner [10], roadside WiFi presents a ubiquitous infrastructure for vehicles to achieve high-quality data transmission in vehicular networks.

To support smooth continuous Internet operation in the presence of dynamics caused by vehicle mobility, we argue that a mobile system that supports accurate real-time identification/localization of roadside WiFi APs is critical; something which current open WiFi community and vehicular devices cannot offer (with errors in tens of meters). Since APs are deployed in a dynamic and unregulated manner, the efficient in-network lookup of roadside APs is key to mobile vehicles seamlessly finding WiFi connections. The desired service must support multiple functionalities including lookup of APs, identification of APs in near range, their number and location; the design of such system presents unique opportunities and challenges.

The AP lookup feature has multiple benefits in roadside WiFi networks. For example, in conditions where a popular AP is congested, the mobile vehicle can switch to other candidate

Manuscript received January 14, 2015; revised August 18, 2015 and January 16, 2016; accepted February 16, 2016; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor Y. Bejerano. This work was supported in part by the National Science Foundation under Grant 1059436 and Grant 1063596, the University of California Transportation Center, and the Intel Collaborative Research Institute for Sustainable Connected Cities. An abbreviated version of this paper appeared in the Proceedings of the ACM/IFIP/USENIX Middleware Conference (Middleware), Bordeaux, France, December 8–12, 2014. *(Corresponding authors: Di Wu and Yong Li.)*

D. Wu is with the Department of Computer Engineering, Hunan University, Changsha 410082, China (e-mail: dwu@hnu.edu.cn).

Q. Liu is with the Department of Computer Science, Dartmouth College, Hanover, NH 03755 USA (e-mail: qiang.liu@dartmouth.edu).

Y. Li is with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: liyong07@tsinghua.edu.cn).

J. A. McCann is with the Department of Computing, Imperial College London, London SW7 2AZ, U.K. (e-mail: j.mccann@imperial.ac.uk).

A. C. Regan and N. Venkatasubramanian are with the Department of Computer Science, University of California at Irvine, Irvine, CA 92697 USA (e-mail: aregan@ics.uci.edu; nalini@ics.uci.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2016.2533399

APs in its communication range [11], [12]. Accurate lookup of roadside APs, deployed outdoors or inside buildings, is also an important step towards understanding the topologies and network characteristics of large scale WiFi networks, *e.g.* network density, connectivity, interference properties, etc, in urban areas [13]. Furthermore, the lookup of APs may reveal interesting social aspects of vehicular networks so that mobile vehicles can be involved in location based services [14], [15] and mobile cloud support [16].

However, enabling accurate lookup in dynamic settings is not straightforward.

1) War-driving [17] techniques proposed for AP lookup in mobile scenarios assume relatively low moving speeds. With fast-moving vehicles, lookup results based on fingerprinting WiFi beacons usually yield rough location estimates with errors in tens of meters - this is often due to the fact that only a small number of beacons can be collected by fast-moving vehicles. Accurate location estimates of roadside APs are critical for the mobile vehicle being able to associate with the best-available AP. We require solutions to improve AP location estimates by an order of magnitude in highspeed vehicular networks, given sparse signal collection opportunity.
2) High mobility in vehicular networks also impacts the connection between mobile vehicles and roadside APs [18]. Ideally, vehicles can obtain a list of APs from the server along its path – such lookup results from existing war-driving databases, *e.g.* Skyhook [19], are simplistic and error-prone since the server side lacks efficient methods to evaluate the accuracy of the information contributed by various mobile users from the same geographic area. Meaningful learning schemes are needed to generate more accurate lookup results by fusing multiple estimates and inferring the overall reliability of each mobile user.

To address the above challenges, we propose `CrowdWiFi`, a crowdsensing system (See Section III) specifically designed for vehicular networks. It consists of two major components to enable efficient lookup on roadside WiFi networks: an online compressive sensing component and an offline crowdsourcing component. The online compressive sensing component running at the vehicle end coarsely localizes nearby APs in real-time while driving, using sparse signal collection capabilities. The Offline crowdsourcing component running at the server end assigns online compressive sensing tasks to some mobile vehicles, then aggregates the online sensing results uploaded by these vehicles, and produces a fine-grained estimation of AP distribution.

In-network localization algorithms require a large number of RSS (Received Signal Strength) readings [20]; this is impractical with fast-moving vehicles. We exploit the use of compressive sensing (CS [21]) techniques to reduce complexity since they allow the recovery of sparse signals with far fewer noisy measurements than that predicted by the Shannon-Nyquist sampling theorem. Unlike several existing solutions [22], [23], we aim to provide an online CS scheme to recover sparse signals by reading and handling dynamic amounts of noisy measurements at runtime in vehicular networks. Feng *et al.* [22] used CS to localize only *one* mobile target from multiple stable reference nodes, which is vastly simpler than the problem we attempt to solve, which requires looking up *multiple targets* from a single mobile vehicle.

Our scheme implements CS using ℓ_1-minimization [21], which can be solved in polynomial time, to online localize APs in a sparse network. We apply our CS scheme in a situation where RSS values are recorded by an *RSS-collector* online, and the number and location of APs must be recovered immediately based on only a few noisy RSS readings and adaptive CS operations, so that efficient estimations with less computation cost and energy consumption can be made in the presence of network dynamics. Upon receiving the measurements, the number of nearby APs can be recognized from their node identifiers (*e.g.* the extended service set identification (ESSID)in 802.11b networks) encapsulated in the measurements (*e.g.* beacon message), and the coarse-grained locations of these APs can be recovered on a grid by CS operations of the RSS values, which are also included in these measurements.

Crowdsourcing [24] refers to the outsourcing or sharing of tasks among loosely defined resources, typically workers, *crowd-vehicles*, in our case. `CrowdWiFi` uses geographical participation allowing servers to assign AP lookup tasks to crowd-vehicles to run the online CS component, and gain information from aggregated answers. A major problem of this crowdsoucing scenario is that the answers are often unreliable and diverse, mainly because it is difficult to monitor the performance of a large collection of crowd-vehicles under various communication environments and mobility situations. In the extreme, there may exist "spammers", who submit random rather than good-faith answers.

Efficient aggregation methods should take into account the differences in the reliabilities of crowd-vehicles' answers. A common strategy to improve aggregation is to add redundancy. `CrowdWiFi` uses a bipartite graph to assign each task to multiple workers and then aggregate the resulting answers. In addition, we address offline crowdsourcing by transforming the aggregation problem into an iterative inference problem on the graphical model, and obtain the reliability of each crowd-vehicle. The *reliability* value presents a probabilistic view of the trustworthy level of a crowd-vehicle on performing crowdsourcing tasks. Therefore, crowd-vehicle with a higher reliability can present more accurate estimation of AP locations. In `CrowdWiFi`, the reliability information is used to refine the estimation of APs using weighted centroid processing. The crowdsourced result can be further used for WiFi topology analysis, or downloaded and shared by other vehicles that will move into these road segments and need Internet access, data dissemination or other infrastructural supports.

To the best of our knowledge, `CrowdWiFi` is the first mobile system using the concept of crowdsensing to localize roadside APs in vehicular networks. The rest of this paper is organized as follows. Section II presents the related work on localization. Section III gives a system overview of `CrowdWiFi`. Section IV describes the online compressive

sensing in CrowdWiFi. Section V addresses the offline crowdsourcing issues in CrowdWiFi. Section VI evaluates CrowdWiFi performance using simulations and real testbed experiments. Section VII concludes our paper.

II. Related Work

In this section, we summarize the most relevant existing research on the two research problems: RSS-based Localization and Crowdsourcing-based Localization.

A. RSS-Based Localization

Because RSS is easy to collect in wireless environments and has no extra hardware requirements, many researchers have explored different in-network localization solutions based on RSS information [25]. A grid-based target lookup algorithm [26] uses a gaussian mixture model and expectation-maximization method to derive the location of wireless sensors, by enumerating probabilities associated with each estimated grid point. Multidimensional scaling (MDS) [27] was designed to analyze the dissimilarities between pairs of WiFi APs from radio scans, and produces a geometric configuration of WiFi APs. Place Lab [28] presents a ranking scheme to sort RSS collected from Wardriving, then apply k-nearest neighbor (KNN) based fingerprinting to localize WiFi infrastructure. Other research activities have been carried out to design robust RSS-based localization algorithms using probabilistic models and calibration enhancements [29], [30]. Compressive sensing based localization has been addressed in [22] and [23] for sparse target estimation via ℓ_1-minimization program.

B. Crowdsourcing-Based Localization

In recent years, research on crowdsourced WiFi fingerprint localization systems has been attracting much attention. Zee [31] is an indoor localization system that makes the calibration zero-effort, by enabling training data to be crowdsourced without any explicit user effort. FreeLoc [32] can extract accurate indoor fingerprint values from short RSS measurement times and achieve calibration-free positioning across devices in crowdsourcing based systems. Zhu *et al.* [33] propose a crowdsourcing localization system that uses both WiFi scene analysis and Bluetooth beacons to address the issues in the bootstrapping stage. The accuracy of crowdsourced answer can be improved via redundancy, such as assigning each task to multiple workers, and aggregate the workers' answers by some method such as majority voting [34]. This problem also can be addressed by building probabilistic models to assign tasks and process answers using standard inference tools. Karger *et al.* [35] give a message-passing style algorithm for deciding which tasks to assign to which workers and for inferring correct answers from the workers' answers.

Note most RSS-based localization approaches have been primarily designed for localizing wireless client nodes, and not the infrastructure. In this paper, we consider the opposite – localization of infrastructure nodes (APs). This is especially useful for roadside WiFi lookup, because of the dynamic nature of APs. In addition, due to the sensing capabilities of fast-moving vehicles, we use crowdsourcing to achieve more accurate lookup results, where we focus on analysis of crowd-vehicles' reliabilities and effective aggregation models for fine-grained estimation of roadside APs.

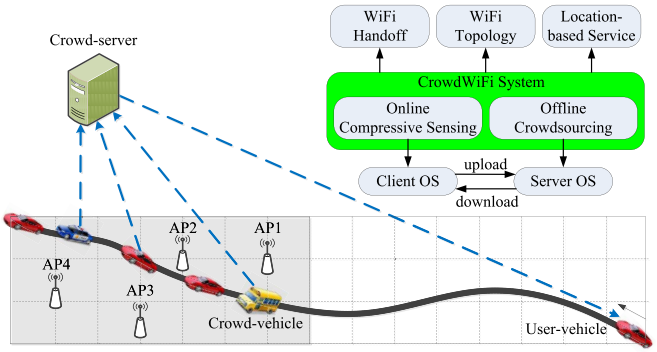


Fig. 1. Crowdsensing system in the lookup of roadside WiFi.

III. System Overview

CrowdWiFi uses node identifiers (*e.g.* ESSID in 802.11b networks) to look up the number of roadside WiFi APs, and consists of two components to look up the locations of these APs: online compressive sensing which quickly localizes nearby APs in a coarse-grained way along the driving route; and an offline crowdsourcing component which evaluates the reliability of each crowd-vehicle after aggregating sensing results producing a fine-grained estimation of AP distribution.

As shown in Fig. 1, CrowdWiFi lies between the operating system (OS) layer and application layer in a distributed computing system. Its online compressive sensing component is running on the client/vehicle OS, and its offline crowdsourcing component runs on the server OS. Through interactive operations between client OS and server OS, mobile vehicles can upload the coarse-grained online sensed results to the server, and also download the fine-grained offline crowdsourced results from the server; therefore the online and offline components are closely related as an organic crowdsensing system. The AP lookup results generated by CrowdWiFi system can provide service support to many software applications in vehicular networks; *e.g.* WiFi handoff, WiFi topology analysis, and other location-based applications.

Three crowdsensing parties are actively involved in CrowdWiFi AP, as shown in Fig. 1, with following specific functions:

- **Crowd-vehicle**: this party plays the role of worker in CrowdWiFi. Crowd-vehicles can be public transportation, e.g. bus, refuge vehicles, patrol cars; that have a regular driving route and schedule, that can provide roadside WiFi sensing services over a given geographical area. Private cars can also upload roadside WiFi data to the crowd-server and be rewarded.
- **Crowd-server**: this component assigns the AP lookup tasks to crowd-vehicles to run online compressive sensing in road segments, and then aggregates crowdsensed answers with different reliabilities. The refined result can

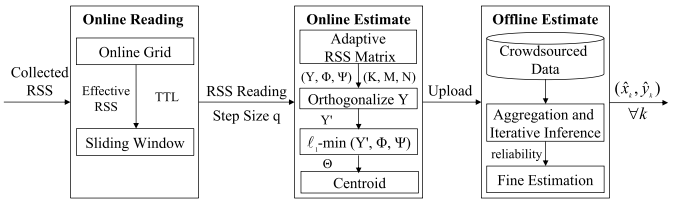


Fig. 2. Workflow (upload case) in CrowdWiFi system.

be further used for WiFi topology analysis, or downloaded by user-vehicles that require WiFi access in those road segments.

- **User-vehicle**: this downloads fine-grained AP lookup results from the crowd-server in advance, and uses this information for opportunistic connection to nearby WiFi APs.

The workflow of `CrowdWiFi` system is illustrated in Fig. 2. We use the upload case from the crowd-vehicle end to the crowd-server end to explain its operational steps. In order to enhance the processing speed and derive the location of APs while the vehicle is moving, we propose an iterative approach for online CS, based on a sliding window with additional steps over the RSS data series collected on a driving grid. The adaptive RSS matrix can further reduce the computation cost and energy consumption by online operations. Through online CS, nearby APs will be located at grid points using coarse-grained estimation. Recall crowd-vehicles data is of variable reliabilities, due to the communication environments (*e.g.* signal interference, malicious attack), and processing capabilities (*e.g.* CPU, memory). Besides assigning *lookup tasks* to crowd-vehicles for online CS, the crowd-server also assigns *mapping tasks* to crowd-vehicles in a bipartite graph to aggregate AP lookup results, and analyze the reliability of each crowd-vehicle using an iterative inference approach. Finally, reliability information is used to refine the estimation of APs via weighted centroid processing. Components of the system model in Fig. 2 will be further explained in details in the following sections.

IV. Online Compressive Sensing

We formulate roadside AP lookup tasks on crowd-vehicles as a compressive sensing problem and propose corresponding online strategies for efficient estimates in `CrowdWiFi`.

A. Compressive Sensing in Roadside AP Lookup

1) Fundamentals of Compressive Sensing: Recently, research has shown that CS can reconstruct a sparse signal with a much lower sample rate than in the Nyquist's Shannon sampling theorem. Let s be a $N \times 1$ column vector. Given an $N \times N$ orthogonal basis $\Psi = [\Psi(1), \Psi(2), \ldots, \Psi(N)]$ where each $\Psi(i)$ being a column vector, s can be expressed by: $s = \Psi\theta = \sum_{i=1}^{N} \theta_i \Psi(i)$, where θ is the sequence of coefficients needed to represent s in the domain of the basis Ψ. Signal s is k-sparse if it is a linear combination of k basis vectors. If $k \ll N$, compressive sensing aims to reconstruct s by

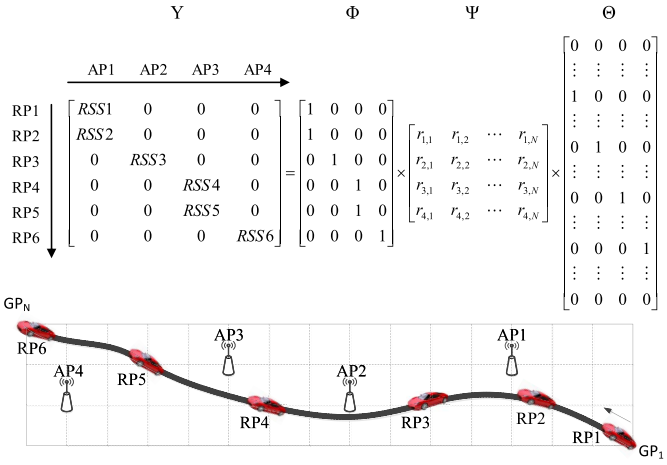


Fig. 3. Compressive sensing in roadside WiFi networks.

taking a set of measurements M much smaller than N by finding a minimal solution to: $y = \Phi s = \Phi\Psi\theta = A\theta$, where y is an $M \times 1$ vector, $k < M \ll N$, Φ is an $M \times N$ measurement matrix, and A is an $M \times N$ matrix.

- ℓ_1-minimization: For a $N \times 1$ vector θ, its solution is obtained from the following ℓ_1-minimization [21], which can be solved in polynomial time, to reconstruct the sparse signal: $\hat{\theta} = \arg\min_{\theta \in A^N} \|\theta\|_1$, s.t. $y = A\theta$, where $\|.\|_1$ is the ℓ_1-norm. If the measurement y include some noise ε, *e.g.* additive white Gaussian noise, then the ℓ_1-minimization to reconstruct θ is $\hat{\theta} = \arg\min_{\theta \in A^N} \|\theta\|_1$, s.t. $y = A\theta + \varepsilon$.

ℓ_1-minimization can be used to recover θ exactly if θ is k-sparse, or compute an approximation of θ that is at least as good as if it is computed from the values and locations of the k most significant coefficients of s.

2) Problem Formulation: Fig. 3 illustrates an AP lookup example with $K = 4$ APs deployed in a vehicular area divided into a discrete grid with $N = 64$ grid points (GPs). While the number of these APs can be recognized from their node identifiers, the locations of these APs are unknown to a mobile vehicle. On-board wireless devices take the drive-by RSS measurements from these APs at $M = 6$ arbitrary reference points (RPs) that are located at 6 grid points over the grid. The RSS collection area around a grid point set for a RP is usually very small (less than $50m^2$) in practice, if we want to achieve accurate AP estimation using the RP location. Considering the moving speed of a vehicle is much higher than others in mobile scenario, it results in a very short time slot (nearly 1 second) to collect RSS measurements from sparse APs (in transmission range) while the vehicle is passing corresponding single RP location. In such a short collection period in vehicular networks, vehicle only can receive one RSS measurement in most cases. For rare exception that a vehicle scans more than one RSS at a RP from different APs, the vehicle should only record one RSS measurement with less noise so that these reliable RSS measurements can be used to derive accurate AP locations.

In highly dynamic and noisy vehicular networks, given above fact that only a small number of RSS measurements

can be recorded in moving vehicle, the goal is to determine the locations of these APs efficiently using the sparse signal collection opportunities. Since $K \ll N$, and the number of measurements $M \ll N$, the AP lookup problem can be well formulated as a sparse matrix recovery problem in the discrete spatial domain using compressive sensing as follows:

$$Y = \Phi\Psi\Theta + \varepsilon, \tag{1}$$

where:

- $Y_{M\times K}$ includes the compressive noisy RSS measurements from K APs and collected by the mobile vehicle on M RPs. Each row vector indicates one measurement value, since the vehicle only records one RSS measurement that has the strongest signal at a short time slot when it is passing corresponding RP location. The number of measurements obeys $M = O(K\log(N/K))$, with $M \ll N$.
- $\Phi_{M\times K}$ is the measurement matrix to record the locations of the vehicle to collect the RSS. Vehicle can know its location from widely used GPS or other positioning systems. Only a small number of RSS measurements are collected by the mobile vehicle over several arbitrary grid points, referred as RPs. Thus, each row of Φ represents the location of each RP, with an element of 1 to indicate the grid point at which the RP is located.
- $\Psi_{K\times N}$ is the sparsity basis, under which the measured signals have sparse coefficients Θ. $[\Psi]_{i,j} = r_{i,j}$ indicates the average value of RSS records sent from AP i and received at grid point j, for all $1 \le i \le K$, $1 \le j \le n$. These RSS records were collected by crowd-vehicles that were previously presented at grid point j (RPs). Other vehicles can download their average value from crowd-server to construct $\Psi_{K\times N}$. The strategy to derive $r_{i,j}$ is explained in Section IV-B3.
- $\Theta_{N\times K}$ is a $N \times K$ matrix denoting the locations of the APs over the grid, and $\Theta = [\theta_1, \theta_2, \ldots, \theta_K]$. Each θ_k is an $N \times 1$ vector with all elements equal to zero except $\theta_k(n) = 1$. n is the index of the grid point at which the kth AP is located.
- ε is the measurement noise.

The specific application of above CS operations is illustrated in Fig. 3 by the matrix Y, Φ, Ψ, and Θ. Note that APs and RPs sometimes are not exactly located at grid points. For the conveniency of CS computing, we select the locations of their closest grid points by Euclidean distance to represent them in `CrowdWiFi`.

In addition, incoherence between the measurement matrix Φ and the sparsity basis Ψ is another important property that should be satisfied to enable the use of CS theory for sparse signal recovery from a small number of measurements [36]. As indicated in [21], the smaller the coherence, the fewer the number of samples needed by CS. However, Φ and Ψ formulated in Fig. 3 are in general coherent in the spatial domain, which violates the incoherence requirement for CS theory. In CS, the Restricted Isometry Property (RIP) provides a sufficient condition for robust recovery of a sparse signal from a small number of noisy measurements. An equivalent description of the RIP is to say that the columns of matrix Φ Ψ should be nearly orthogonal [21]. Therefore, we propose the following orthogonalization preprocessing procedure that restores incoherence properties to facilitate compressive sensing.

Assume the noisy measurement matrix $Y = [y_1, y_2, \ldots, y_K]$ with size $M \times K$. $Y = \Phi\Psi\Theta + \varepsilon$, where $\Theta = [\theta_1, \theta_2, \ldots, \theta_K]$. Each column of Θ, namely θ_k $(1 \le k \le K)$, is an $N \times 1$ vector with all elements equal to zero except $\theta_k(n) = 1$, where n is the index of the grid point at which the k-th AP is located; each θ_k is a 1-sparse vector. Since CS theory deals with reconstructing unknown vectors, we can obtain Θ by applying CS to each θ_k separately. Given an online RSS reading for the k-th AP in y_k, corresponding θ_k can be well reconstructed with orthogonal operations in Proposition 1.

Proposition 1: Let $y'_k = Ty_k$, $A = \Phi\Psi$, where $T = QA^\dagger$ and $A^\dagger$ is a pseudoinverse of matrix A. Also let $Q = \text{orth}(A^T)^T$, where $\text{orth}(A)$ is an orthogonal basis for the range of A, and A^T returns the transpose of matrix A. If $M \ge O(K\log(N/K))$, then θ_k can be well recovered from y'_k via an ℓ_1-minimization program.

Proof: See Appendix A. ■

B. Online Localization of Roadside AP

While in theory ℓ_1-minimization is solvable in polynomial time [21], it becomes computationally expensive when the number of grid points N is large. When a vehicle moves, it continuously reads the RSSs from different APs, so the size of noise measurement matrix Y increases fast online, as shown in Fig. 3. Meanwhile, the size of Φ and Ψ will be updated as well in order to get an online estimate of Θ.

In order to further enhance the processing speed and derive the location of the wireless APs while the vehicle is moving, we propose an online CS approach in `CrowdWiFi`, using a sliding window and an iteration step over the collected RSS data series. In comparison with traditional CS localization using orthogonal noisy RSS measurements and ℓ_1-minimization as introduced in Section IV-A.2, the proposed online CS in `CrowdWiFi` applies additional steps to online localization based on limited RSS measurements, and can achieve low cost CS computation for efficient estimation through several iterations. The workflow of online CS is illustrated in Fig. 2.

1) Grid Formation: Vehicle trajectories typically cross a large geographic area resulting in a large number of grid points and large sparsity bases Ψ. To reduce irrelevant or redundant information, the online AP lookup from current grid points in the reachable region is used.

Online grid formation in `CrowdWiFi` is based on the dynamic time-dependent routes in each round of online CS. In the n-th round of `CrowdWiFi` with inputs R_n, we localize the APs using the grid structure of the current driving area. We derive the boundaries of the driving area according to the input RSS measurements R_n and their corresponding RP location information. Simply, the boundaries of the fixed area is defined by a rectangle with $(x_{\min} - T_m, y_{\min} - T_m)$ and

$(x_{\max} + T_m, y_{\max} + T_m)$ as the lower-left and upper-right corners' coordinates, and $x_{\min}$, $y_{\min}$ and $x_{\max}$, $y_{\max}$ are the minimum and maximum x and y coordinates of the series of RPs' locations, respectively. T_m is the communication radius of the RSS-collector in the vehicle.

Depending on the accuracy and computation cost of the lookup algorithm, the edge length of each lattice in the grid structure can be determined.

2) Sliding Window Based RSS Reading: In order to further enhance the processing speed in `CrowdWiFi` while the vehicle is moving, we take an iterative approach using a sliding window over the RSS series. Each RSS is tagged with a time stamp and TTL (Time to Live) as it is useless when out of date and should be removed.

As the RSS-collector gathers RSS information, we group the most recently collected RSS data series into a small data set for current estimations of nearby AP location. Suppose the length of the current collected RSS sequence is k. We use a sliding window with a length of $s(s < k)$ to extract the input data sequence from the current collected RSS sequence. The iteration step size is set to $q(q < s < k)$. Then, the set of the input RSS sequences in the n-th round of iteration is $R_n = \{r_{q(n-1)+1}, r_{q(n-1)+2}, \ldots, r_{q(n-1)+s}\}$.

In each iteration, `CrowdWiFi` estimates the likely locations of APs by the CS operation on a grid structure, as explained in Section IV-A2; one credit is granted to each of the estimated locations mapped at the grid points. After several iterations of selecting q RSS values from data set R_n, `CrowdWiFi` consolidates these estimates, by merging credits of aligned location estimates and filtering out spurious estimates with few credits; then refines these APs' location through offline crowdsourcing introduced in Section V-E.

3) Constructing Sampling Matrix: Vehicular network presents a complex environment for signal propagation and could produce signal blocking and some RSS outliers, therefore it is hard for single vehicle to collect complete and reliable RSS samples of roadside APs. A feasible approach is to collect many samples and get their average value in the crowd-server, so that other vehicles can download the average RSS of each AP for online localization.

The sparsity basis $\Psi_{K\times N}$ in Section IV-A.2 is the downloaded sampling matrix, where $r_{i,j}$ are the average value of RSS samples sent by AP i and received by different crowd-vehicles at grid point j. Suppose the total number of RSS samples received from AP i at grid point j is c; then, the corresponding average RSS value is: $r_{i,j} = \sum_{\tau=1}^{c} r_{i,j}(\tau)/c$, where $r_{i,j}(\tau)$ represents the τth RSS value from the AP i for grid point j. Since the received signal power of wireless network (802.11 variants) should range somewhere between -100 to $-10dbm$, if an AP i was not perceived by any crowd-vehicle before at grid point j, then $r_{i,j} = 0dbm$ by default setting in `CrowdWiFi`, meaning no WiFi signal were detected at the grid point before.

In reality, the power status (*e.g.* off) and placement (*e.g.* removal) of roadside APs change over time. Under these circumstances, their previous RSS samples become out of date. In order to provide average value of effective RSS samples for online localization of dynamic roadside APs, each RSS sample in crowd-server is also tagged with a time stamp to indicate its time to collect, and an TTL to indicate its valid period.

4) Centroid Processing: The CS model in Fig. 3 is presented for the case that APs are exactly at grid points. In that case, each recovered $\hat{\theta_k}$ from ℓ_1-minimization is a $N\times 1$ vector with one entry equals to 1, which correspondingly pin-points the grid point at which the relevant AP is located, and all other entries equal to 0.

However, the APs may not be exactly located at these grid points. In such cases, the recovered $\hat{\theta_k}$ does not turn out to be an exact 1-sparse vector, but with a few non-zero entries. These non-zero entries in each $\hat{\theta_k}$ correspond to the grid points whose distance from the AP is less than some threshold, and the values at these entries are inversely proportional to the distance from the AP and normalized to have a sum of one (the case of AP exactly at a grid point can be easily shown as a special case). Therefore, a post-processing procedure by centroid approach is conducted for real applications. We choose the dominant coefficients in $\hat{\theta_k}$ whose values are above a certain threshold ζ, and take the centroid of these grid points as the location indicator. Let $\mathcal{S}_k$ be the set of all indexes of the elements of $\hat{\theta_k}$ such that: $\mathcal{S}_k = \{n|\hat{\theta_k}(n) > \zeta\}$.

These are potential candidate grid points for the estimate of the location of the kth source (actual AP). Each $i \in \mathcal{S}$ represents a grid point in the two dimensional space (x_i, y_i). The estimated location of source k, denoted by $(\hat{x_k}, \hat{y_k})$, can be derived by finding the centroid of the candidate points, denoted by (x_i, y_i), using weighted mean in (2). The weight of each grid point (x_i, y_i) is its corresponding value $\hat{\theta_k}(i)$ computed from the ℓ_1-minimization. We have,

$$(\hat{x_k}, \hat{y_k}) = \frac{1}{\sum_{i\in\mathcal{S}_k} \hat{\theta_k}(i)} \sum_{i\in\mathcal{S}_k} \hat{\theta_k}(i)(x_i, y_i). \tag{2}$$

The above centroid processing running on a moving vehicle provides an online approximation to the actual location from the reconstructed signal. Note that CS theory guarantees that the reconstruction error is proportional to the noise level, where the random, uncorrelated noise could be caused by fluctuations of the RSS, malicious interference, and the fact that the vehicle does not take measurements exactly at grid points. The noise level also reflects the reliability of a crowd-vehicle in `CrowdWiFi` (*i.e.* if the noise level is high, the reliability is correspondingly low). Therefore the AP location results of centroid processing from a crowd-vehicle could be coarse-grained estimations. In order to compensate for the reconstruction errors induced by the diverse reliabilities (noise levels) of different crowd-vehicles, an additional fine-estimation of AP locations is proposed in Section V-E by crowdsourcing the reliability information.

5) Adaptive RSS Matrix: In `CrowdWiFi`, smaller lattice size in the grid structure can produce more accurate location estimation due to its generation of a large number of grid points N. However, large N in sparse network also indicates many redundant grid points where no roadside AP exists or RSS can't be collected (see $\Psi_{K\times N}$ in Fig. 3), therefore

results in extra computation and energy consumption to run CS.

Proposition 2: Given K nearby APs and M RPs in online localization, the energy consumption E for a vehicle to localize these APs is proportional to the number of grid points N, as: $E \propto \lambda N, \lambda = \Omega(KM)$.

Proof: See Appendix B. ■

From Proposition 2, we know that for online energy saving an efficient mechanism is needed to select useful grid points and achieve dynamic construction of RSS matrix. For online but static RSS matrix Y in Fig. 3, we adaptively remove unreliable online RSS to decrease the number of rows, following a fitness function as: $f = Y \times W$, where Y is the static $M \times K$ matrix, generated by traditional CS, with M online RSS measurements; $W = [w_1, w_2, \ldots, w_K]^T$ includes the weighted value w_i to indicate the utilization rate of each sensed AP i, $i \in 1, 2, \ldots, K$. The weighted value can be obtained from the crowd-server through analyzing the offline RSS samples (see Section IV-B.3) collected by crowd-vehicles; it is directly proportional to the number of grid points that received RSS from AP i, defined as: $w_i = N_i/N$, where N_i is the total number of grid points where the AP i is perceived, and N is the total number of grid points in current grid formation. `CrowdWiFi` uses fitness function to rank the M APs by their online RSS value and offline weighted value, and choose top M' APs ($M' < M$) to construct new online RSS matrix Y'. As for new RSS sampling matrix Ψ', once crowd-server initiates a static Ψ as the original $K \times N$ matrix, it removes the columns with all 0 in Ψ to construct an adaptive $K \times N'$ matrix Ψ' ($N' < N$). These removed columns represent the redundant grid points and they have no impact on the estimation of other grid points with RSS records.

Compared to the original matrix Y and Ψ, the adaptive matrix Y' and Ψ' are composed of more reliable RSS measurements and less redundant grid points, therefore can improve the estimation efficiency and reduce the energy consumption.

V. Offline Crowdsourcing

After online roadside WiFi compressive sensing, the crowd-vehicle uploads the geographical distribution of nearby APs, with the number, location and grid formation, to the crowd-server for offline crowdsourcing.

A. Crowdsourcing Tasks

`CrowdWiFi` uses geographical participation allowing crowd-servers to assign two types of crowdsourcing tasks to crowd-vehicles for information aggregation and analysis. The two tasks are defined as follows:

1) Lookup Task: the crowd-vehicle runs online compressive sensing in road segments along its driving route to lookup nearby APs, and then sends the estimations to the crowd-server. The crowd-server usually assigns multiple lookup tasks to each crowd-vehicle to estimate APs in different road segments.
2) Mapping Task: the accuracy of AP lookup results presented by crowd-vehicles in the same road segment varies by the different reliability levels of these crowd-vehicles. Crowd-server enumerates possible distribution patterns (number and location) of APs in a road segment. Once a crowd-vehicle finishes the lookup task in the road segment and submits its answer to the crowd-server, the crowd-vehicle will be asked to map its answer with these distribution patterns, so that the crowd-server can derive the reliability of each crowd-vehicle. The crowd-vehicle with higher reliability can provide more accurate AP lookup results.

`CrowdWiFi` assigns the crowdsourcing tasks to crowd-vehicles using a bipartite graph, and aggregates answers and analyze the reliability of each crowd-vehicle using an iterative inference approach on the graphical model. `CrowdWiFi` also refines the AP locations using these crowdsourced answers. Detailed techniques will be explained in following sections.

B. Spammer-Hammer Model

`CrowdWiFi` assumes that all the lookup tasks have the same level of difficulty, but that crowd-vehicles may have different reliabilities; due to heterogeneous communication environments and processing capabilities, or they are spammers.

We assume the reliability of crowd-vehicle j is measured by a single parameter q_j, which corresponds to its probability of correctness. The values of q_j reflect the diversity of crowd-vehicles' reliability, represented as independent distributed random variables with a given distribution on [0, 1]. One typical example is *spammer-hammer* model, as described in [35], where, $q_j \approx 1$ correspond to *hammers* that provide reliable answers, and $q_j \approx 1/2$ denote *spammers* that give random answers. We assume the q_j of all crowd-vehicles are drawn independently from a common prior $p(q_j|\delta)$, where δ are the hyper-parameters. To avoid the cases where spammers overwhelm the system, it is reasonable to require that $\mathbb{E}[q_j|\delta] > 1/2$. Typical priors in *spammer-hammer* model includes the discrete prior, where $q_j \approx 0.5$ or $q_j \approx 1$ with equal probability.

C. Graphical Model

To aggregate results from unreliable crowd-vehicles and determine their reliability value q_j, we have designed a bipartite graph scheme for the crowd-server to enumerate and assign mapping tasks to crowd-vehicles. As shown in Fig. 4(a), the task assignment can be represented by a bipartite graph where an edge (i, j) denotes that the mapping task i is labelled by the crowd-vehicle j. Each mapping task is a possible distribution pattern (number and location) of APs, denoted by blue dots in the grid, given a road segment ID. Crowd-vehicles submit their lookup answers if these distribution patterns exist (labeling $+1$) or not (labeling -1). Initially, some AP distribution patterns are generated randomly by the crowd-server for bootstrapping. After that, distribution patterns can be added into the mapping set by selecting the lookup results from crowd-vehicles, so that the crowd-server can avoid generating too many non-existent AP distribution patterns; saving computation and assignment cost.

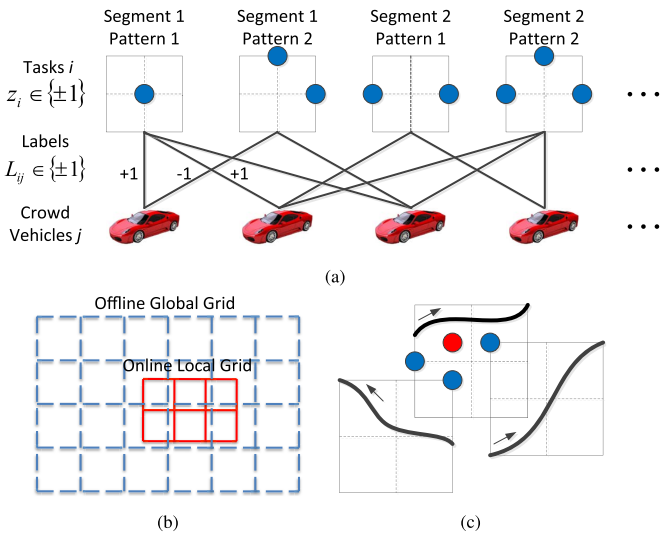


Fig. 4. Aggregation in offline crowdsourcing. (a) Bipartite graph. (b) Grid mapping. (c) Fine-grained estimation.

Specifically, we assume there are M crowd-vehicles and N mapping tasks with binary labels ± 1. The true label of task i is denoted by $z_i \in \pm 1$, $i \in [N]$, where $[N]$ represents the set of first N integers. $\mathcal{N}_j$ is the set of tasks labelled by crowd-vehicle j, and $\mathcal{M}_i$ is the set of the crowd-vehicles labelling task i. The labelling results form a matrix $L \in 0, \pm 1^{N \times M}$, where $L_{ij} \in \pm 1$ denotes the answer if crowd-vehicle j labels task i, and $L_{ij} = 0$ if otherwise. Therefore, the reliability q_j of crowd-vehicle j is defined by the probability of correctness: $q_j = \text{prob}[L_{ij} = z_i]$. The goal is to find an optimal estimator $\hat{z}$ of the true labels z given the observation L, minimizing the average bit-wise error rate $\frac{1}{N} \sum_{i \in [N]} \text{prob}[\hat{z}_i \neq z_i]$.

D. Iterative Inference

Using majority voting to aggregate labels L is error-prone since it weights all the source crowd-vehicles equally. We use an iterative inference approach for `CrowdWiFi`, based on a message-passing algorithm proposed by [35]. Let $x_{i \to j}$ and $y_{j \to i}$ be real-valued messages from tasks to crowd-vehicles and from crowd-vehicles to tasks, respectively. Initializing $y^0_{j \to i}$ randomly from Normal(1, 1) or deterministically by $y^0_{j \to i} = 1$, iterative inference updates the messages at t-th iteration via

$$x^{t+1}_{i \to j} = \sum_{j' \in \mathcal{M}_{i \setminus j}} L_{ij'} y^t_{j' \to i}, \quad y^{t+1}_{j \to i} = \sum_{i' \in \mathcal{N}_{j \setminus i}} L_{i'j} x^{t+1}_{i' \to j}. \quad (3)$$

A crowd-vehicle message $y_{j \to i}$ represents the updated reliability of the crowd-vehicle j, and the labels are estimated via the sum of the answers weighted by each crowd-vehicle's reliability as: $\hat{z}^t_i = \text{sign}[\hat{x}^t_i]$, where $\hat{x}^t_i = \sum_{j \in \mathcal{M}_i} L_{ij} y^t_{j \to i}$. Note that the 0th iteration of iterative inference reduces to majority voting when initialized with $y^0_{j \to i} = 1$.

After transforming the labeling aggregation problem into an iterative inference problem on a bipartite graphical model, the joint posterior distribution of crowd-vehicles' reliabilities $q = q_j, j \in [M]$ and the true labels $z = z_i, i \in [N]$ conditional on the observed labels L and hyper-parameter δ is

$$\begin{aligned} p(z, q | L, \delta) &\propto \prod_{j \in [M]} p(q_j | \delta) \prod_{i \in \mathcal{N}_j} p(L_{ij} | z_i, q_j) \\ &= \prod_{j \in [M]} p(q_j | \delta) q_j^{c_j} (1 - q_j)^{\gamma_j - c_j}, \end{aligned} \quad (4)$$

where $\gamma_j = |N_j|$ is the number of answers made by crowd-vehicle j, and $c_j := \sum_{i \in \mathcal{N}_j} \text{II}[L_{ij} = z_i]$ is the number of j' s answers that are correct. By standard Bayesian arguments, one can show that the optimal estimator of z to minimize the bit-wise error rate is given by

$$\hat{z}_i = \arg \max_{z_i} p(z_i | L, \delta), \quad (5)$$

where $p(z_i | L, \delta) = \sum_{z_{[N] \setminus i}} \int_q p(z, q | L, \delta) \text{d}q$.

E. Crowdsourcing for Fine-Grained Estimation

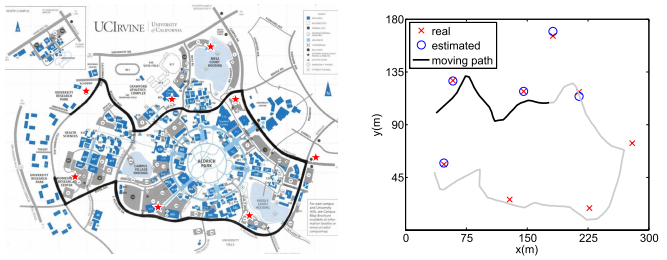
Recall, the online compressive sensing finds the coarse-grained location estimations of APs over the grid structure. Using iterative offline inference, optimal estimations of z and the corresponding reliability of each crowd-vehicle can edge the locations closer to the optimum.

As shown in Fig. 4 (b), the crowd-server maintains an offline global grid structure to cover a large area. When an online local grid structure is formed by a mobile vehicle in a segment of driving route in the area, the online grid will be mapped into the offline grid, by placing its grid points at corresponding positions according to the offline coordination system. By the grid mapping, crowd-server can transfer the coordinate information in `CrowdWiFi` (*e.g.* RSS, AP) between offline grid and online grid for localization.

According to Section IV-B1, different grid structures from local views may form, even though they are in the same sensing area. As shown in Fig. 4 (c), the estimated locations of an AP (estimations could be classified by the AP identifiers) from three crowd-vehicles, denoted by three blue dots, are located at different grid points of three grid formations. The three estimated locations for the same AP are different, due to the three crowd-vehicles with diverse estimation reliabilities moving over different paths. A centroid processing procedure is conducted by the crowd-server to further improve the AP's estimation by merging its three estimates to one, denoted by the red dot in Fig. 4 (c). A crowd-vehicle with a higher reliability can present more accurate estimation, therefore the centroid is weighted based each vehicle's reliability accordingly, similar to the centroid method illustrated by Eq. 2 in Section IV-B4.

F. Crowdsourcing Platform

The crowdsourcing platform in `CrowdWiFi` consists of four components: (1) a web interface where vehicle can upload and download nearby APs information based on its location; (2) a crowd-server including a database for storing the crowdsourced APs information, and distributing the information to potential users; (3) a client application for the



(a) (b)

Fig. 5. Trajectory illustrations of APs lookup in UCI scenario. (a) Simulation map. (b) 60 data points.

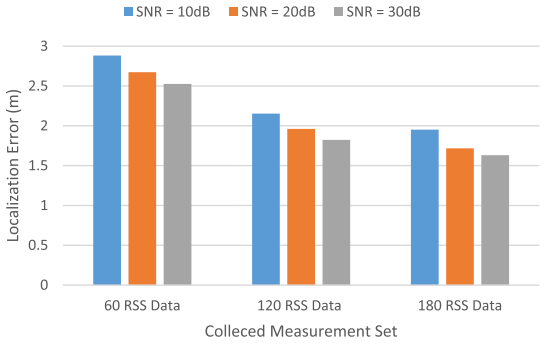


Fig. 6. Localization error *vs.* measurement noise.

crowd-vehicle, which pulls available AP lookup tasks from the database, based on its given driving route and geographical information; and (4) a client application for user-vehicles to obtain AP information. Through this crowdsourcing platform, `CrowdWiFi` is able to provide nearby AP information to vehicles requesting wireless access and data dissemination.

The driving routes and locations of crowd-vehicles, their AP lookup results, and other relevant crowdsourcing factors are privacy information. Therefore, in `CrowdWiFi`, crowd-vehicles have the right to accept tasks to share these information for rewards, or deny the tasks to protect their privacy.

VI. Performance Evaluations

`CrowdWiFi` is evaluated in terms of lookup accuracy by simulation and real testbed experimentation. To define the localization error, for each grid i, let K be the number of roadside APs. Assume there are K actual APs with locations $\{(x_i, y_i)\}, \forall i = 1, 2, \ldots, K$, and there are corresponding K estimated APs with locations $\{(\hat{x_i}, \hat{y_i})\}$. Thus we derive the localization error as the normalized relative distance: $(\sum_{i=1}^{K} \sqrt{(x_i - \hat{x_i})^2 + (y_i - \hat{y_i})^2})/(Kl)$, where l is the length of grid lattice. If the error is less than 100%, then it indicates that the estimated location is close to the real location as their distance is shorter than the grid diameter.

A. Simulation Results

Eight campus AP deployments in the University of California, Irvine (UCI) were used and RSS values from periodic radio packets were collected along the path shown in Fig. 5(a). We used NCTUns v5.0 [37] to model the vehicular network and scaled the UCI campus map over a $300m \times 180m$ rectangular area in our simulations. The distance between each pair of APs is more than $50m$, and the effective signal transmission radius of the APs is $100m$. The lattice size in grid structure is set as $8m \times 8m$.

In order to demonstrate the AP lookup performance using online CS, we first set the 8 APs to be physically located at 8 grid points and then run online CS to estimate locations at three different moments: those when the RSS-collector collected the 60-th; 120-th; and 180-th RSS values in the simulation respectively. The resulting location estimates of the 60 RSS values are shown in Fig. 5(b), where the bold dark curve shows the online trajectory, the crosses represent the actual AP locations, and circles are location estimates from `CrowdWiFi`. The sliding window size for each computation iteration is set as 60, and the iteration step size is 10. That is, we re-run `CrowdWiFi` using the past 60 data samples when the RSS-collector collects an additional 10 RSS values.

The signal-to-noise ratio (SNR) is commonly used in wireless communication as a way to measure the quality of wireless connections. If data transmitted to the receiver are to be received correctly, a sufficient SNR must be maintained at the receiver. In order to test the robustness of our algorithm, we intentionally add Gaussian white noise to the observation vector y. The average localization error of with respect to the measurement noise, running in the scenario as described in Fig. 5, is presented by Fig. 6. The SNR is changed from 10dB to 30dB to evaluate its impact on our online CS based localization when 60, 120 and 180 RSS data are collected.

As Fig. 6 shows, when vehicles move around the amount of RSS readings increase, and our online CS operations can provide accurate estimates of AP locations. In comparison with the 60 RSS data set, the 120 data set and 180 data set can achieve more accurate location estimates. Based on the three data sets for location estimate, the average estimate error reduces from 2.53 meters (for the 60 points) to 1.63 meters (for the 180 points) when SNR $=$ 30dB; satisfactory error bounds. It also shows the online CS approach can achieve certain localization accuracy under various SNR. When the number of measurements conforms with the CS theory, the localization error is below 2 meters for the 180 RSS data set; $1.95m$ for SNR values at 10dB, $1.72m$ for SNR values at 20dB, and $1.63m$ for SNR values at 30dB. The convergence trend of localization error confirms that our online CS-based localization can tolerate a certain level of noisy RSS measurements, and effectively reduce the impact of SNR on RSS collection and processing. Therefore, it suits the harsh localization environment in vehicular networks. For performance evaluation, we set SNR=30dB in following simulation experiments.

The lattice size (number of grid points) is an important parameter for CS, and can determine `CrowdWiFi`'s accuracy level. Fig. 7(a) examines the impact of lattice size on localization error over 180 data points. The localization error is computed by the above mentioned method times 100%. Fig. 7 shows that a smaller lattice length (larger number of grid points) results in more accurate location estimation. When the lattice is less than or equal to 10 meters, `CrowdWiFi` achieves

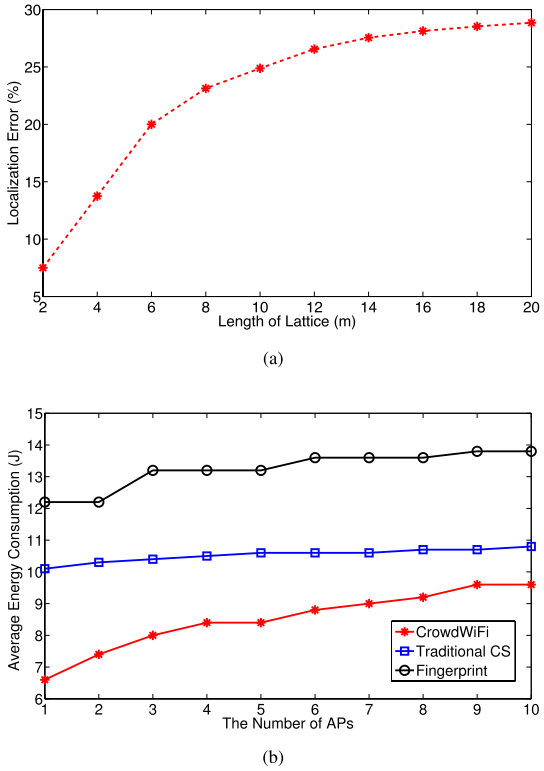


Fig. 7. Impact of grid structure in CrowdWiFi. (a) Lattice size. (b) Energy consumption.

a low estimation error of less than 2 meters. When the lattice length is around 20 meters, CrowdWiFi can maintain the error at a level below 3 meters.

Intuitively, while the estimation error increases with the increased lattice length, a very small lattice length will result in more computation costs and energy consumption due to the increased number of redundant grid points. As shown in Fig. 7(b), given the lattice length as 5 meters, CrowdWiFi has a lower average energy consumption compared to the traditional CS [22] and fingerprint [28] with respect to different number of APs, because it can adaptively choose useful grid points to reduce the size of RSS matrix in the CS operation. The dynamic selection scheme is not well considered in the traditional CS and fingerprint, therefore, when the average energy consumption of these two algorithms ranges from $10J$ to $13J$, CrowdWiFi's average energy consumption is below $10J$, which exhibits significant energy efficiency. According to the results, we learn that choosing a feasible lattice length between 5 to 10 meters can guarantee satisfactory localization performance.

Since online CS provides estimates of APs located at grid points, and in reality most APs are not exactly located at grid points, the true locations of 8 APs are set randomly on the grid structure for the second set of simulations. Then we verified the performance of offline crowdsourcing on fine-grained estimation.

We generated the reliabilities q_j from the spammer-hammer priors, that equal 0.5 or 1.0 with certain probabilities. The assignment graphs were randomly drawn from the set of (ℓ, γ)-regular bipartite graphs with 1000 tasks. The left degree ℓ denotes the number of crowd-vehicles per task, and the right γ denotes the number of tasks per crowd-vehicle. For comparison, we also calculated majority voting (MV) to chooses what the crowd-vehicles decide [34], Skyhook that compares relative rankings using the Spearman rank-order correlation coefficient [28], and the oracle lower bound (Oracle) that assumes the true reliability values q_j of all crowd-vehicles are known. We terminate all the iterative algorithms at a maximum of 100 iterations or with 10^{-6} message convergence tolerance. All results are averaged on 100 random trials. As shown in Fig. 8 (a) and (b) in our experiments, the bit-wise error rate of the iterative inference in CrowdWiFi is shown to be lower than majority voting and Skyhook due to its reliability evaluation, and CrowdWiFi scales in the same manner to approach oracle lower bound when the number of workers per task and the number of tasks per worker increase. Also, we show that the error rates of the crowdsourcing algorithms generally decay exponentially w.r.t. the degree ℓ and γ of the assignment graph on a spammer-hammer model.

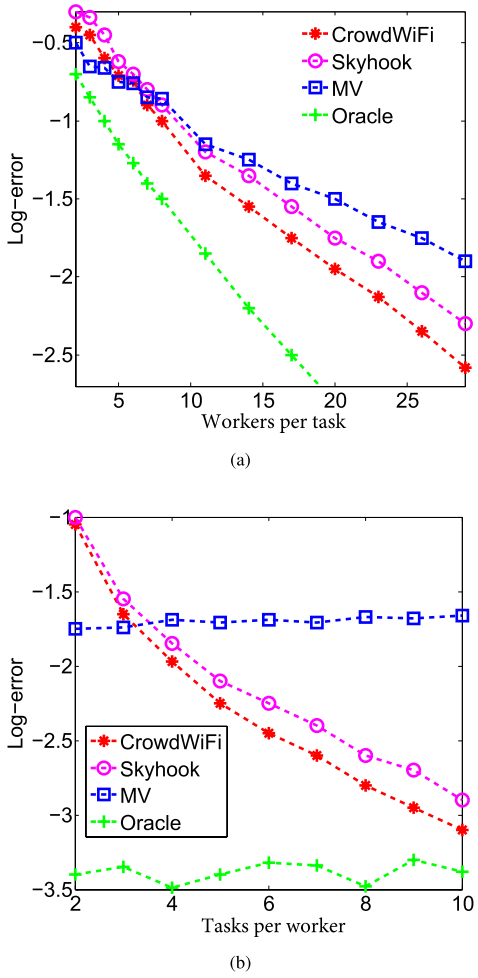


Fig. 8. Performance of crowdsourcing in bipartite assignment. (a) ℓ(fixed $\gamma = 5$). (b) γ(fixed $\ell = 15$).

The performance of offline crowdsourcing combined with online CS on AP lookup is examined in the next simulations. We compare CrowdWiFi with two specific state-of-art algorithms: the Gaussian mixture model based grid algorithm [26],

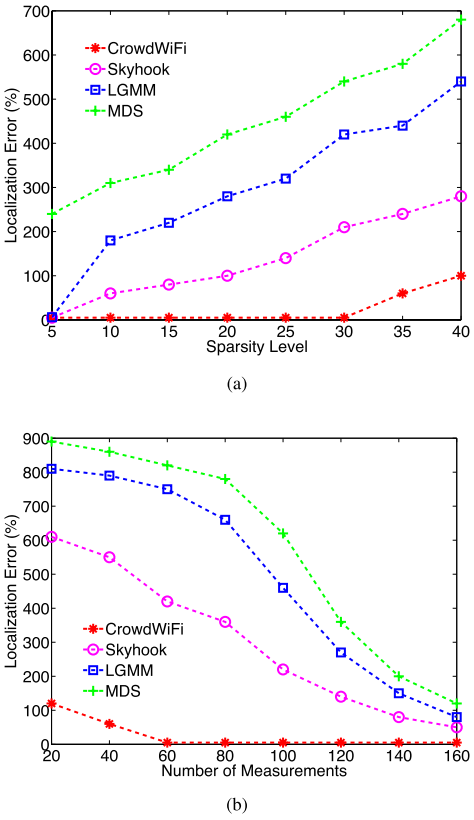


Fig. 9. Comparisons between CrowdWiFi and other algorithms on localization errors. (a) Localization *vs.* sparsity level. (b) Localization *vs.* measurement.

and the multidimensional scaling (MDS) based radio scan algorithm [27], referred to as "LGMM" and "MDS", respectively. We also implemented "Skyhook" based on Place Lab's fingerprinting algorithm [28] (Skyhook's wardriving algorithm is proprietary, but similar to Place Lab). The localization error is computed by above mentioned method times 100%, respectively. The simulation area is $250m \times 250m$, and the lattice size in grid is set as $8m \times 8m$.

Fig. 9(a) depicts the localization error *vs.* the sparsity level K when $N = 900$, $M = 160$, where K, N and M indicate the number of WiFi APs, the number of grid points and the number of reference points, respectively. We also set SNR = 30dB. It can be observed that CrowdWiFi and Skyhook exhibit much lower error than other algorithms due to the adoption of crowdsourcing. In addition, because CrowdWiFi employs compressive sensing for sparse signal processing and reliability based crowdsourcing, it can achieve better performance than Skyhook. When $K = 30$, the localization errors is almost zero, while other algorithms produce a localization error of over 200% under the same scenario. Even when K is as small as 10, the localization error of LGMM, MDS, and Skyhook is still above 60%. When $K = 40$, CrowdWiFi achieves a localization error of less than 100% (inside a grid) while other algorithms result in a much lower accuracy.

Fig. 9 (b) depicts the localization error *vs.* the number of measurements M when $N = 900, K = 10$ and SNR = 30dB. As an overall trend, the larger the M, the smaller the error for all algorithms. Note that the localization error for

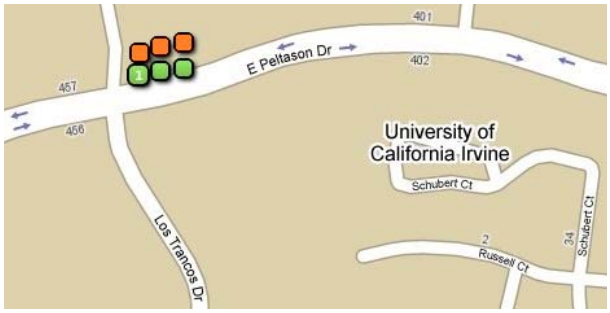


Fig. 10. Real testbed for online lookup of roadside APs.

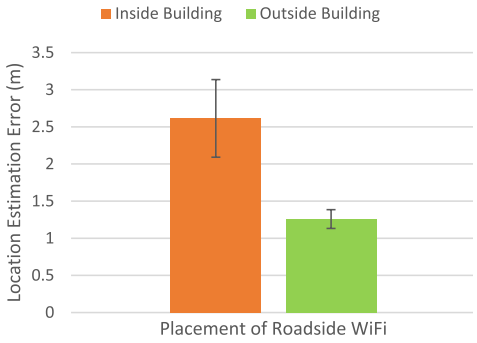


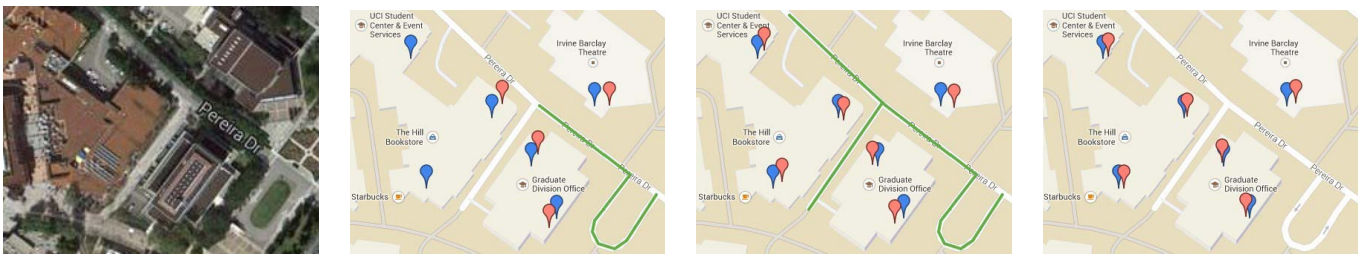
Fig. 11. Localization error *vs.* AP placement.

CrowdWiFi are almost zero when $M \geq 40$, while other algorithms yield much higher errors when $M < 100$. This clearly indicates that CrowdWiFi does not require a large number of measurements to precisely estimate the number and location of the APs and is therefore highly practical.

B. Testbed Experiment Results

In our real testbed experiments, we used Open-Mesh wireless mesh nodes running IEEE 802.11b/g as APs [38]. Each inexpensive node is an integrated access point, mesh gateway and repeater, though our work can apply to other kinds of roadside APs. The transmission radius of the Open-Mesh nodes is approximately 30 meters. As the RSS-collector, a ThinkPad X61 Laptop with Intel(R) PRO/Wireless 3945ABG Network Connection was used to collect the RSS values along the moving path of our vehicle.

In the first set of experiments, we evaluate the performance of online CS, and how location estimations are affected when the roadside APs are located inside buildings (behind walls from the roads) in comparison with the APs outside buildings. We deployed three Open-Mesh nodes inside and three Open-Mesh nodes outside the California Institute for Telecommunications and Information Technology (Calit2), which is a building beside the E Peltason Dr at UCI, as roadside APs. The APs inside Calit2 building are labelled by orange icons, and the ones outside building are labelled by green icons, as shown in Fig. 10. A vehicle was moving at speed around $15mph$ (miles per hour) back and forth on a road section of the E Peltason Dr, as labelled by the arrows on the road in Fig. 10, to collect RSS samples from the six APs and estimate their locations. The lattice size of the grid structure is set as $10m \times 10m$. The sliding window size for each computation



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

Fig. 12. Roadside APs lookup and crowdsourcing in UCI campus testbed experiments. (a) Testbed map. (b) 20 data points, 20mph (estimation error 3.85m). (c) 40 data points, 20mph (estimation error 3.31m). (d) Crowdsourced result (estimation error 2.04m).

iteration in online CS is set as 20, and the iteration step size is 10. The experiments run around 220 seconds.

Fig. 11 presents the average location estimation error and its standard deviation for different placements of roadside APs. The average localization error is 2.61m for APs placed inside Calit2 building, and 1.26m for APs placed outside building. In addition, the estimation deviation of placement inside building is bigger than outside building. The results are caused by the collected RSS measurements that fluctuate depending on the propagation environment. When vehicle moving on the road collects RSS measurements from roadside APs (in transmission range) placed inside building, these signals come much more background noise and interference than outside scenario because of the influence of indoor structure. However, these online errors from indoor environment can be effectively compensated by our offline crowdsourcing based fine-grained localization scheme in `CrowdWiFi`; we will illustrate the effects by the second set of real experiments as follows.

In the second set of real-world experiments, we evaluate the performance of our crowdsourcing based fine-grained localization scheme in `CrowdWiFi`. We deployed six Open-Mesh nodes at six different locations: two in the Graduate Division Office, one in Irvine Barclay Theatre, one each in The Hill Bookstore, Starbucks and the UCI Student Center, over a 100×100 square meters area on UCI campus as shown in Fig. 12. The blue icons indicate the real locations, the red icons indicate the estimated locations, and the green solid line indicates the moving path of our vehicle. We chose a different campus scenario here because it presents some APs placed further away from the moving path of our vehicle, in comparison with the placement of roadside APs in Fig. 10, therefore `CrowdWiFi` can be tested in a harsher localization environment to evaluate its crowdsourcing performance.

For AP lookup by crowdsourcing, the vehicle collected RSS values from nearby APs at three different average moving speeds: 10mph, 15mph, and 20mph. We present the location estimation results for each moving speed at two different moments of the experiment; those when the RSS-collector collected 20-th and 40-th RSS samples, as shown in Fig. 12(b) and Fig. 12(c), for the case that the vehicle is moving at speed 20mph. The offline crowdsourcing platform aggregated the AP lookup results of the three moving speeds, and obtained a corresponding reliability measure of the crowd-vehicle. After centroid processing of the crowdsourced

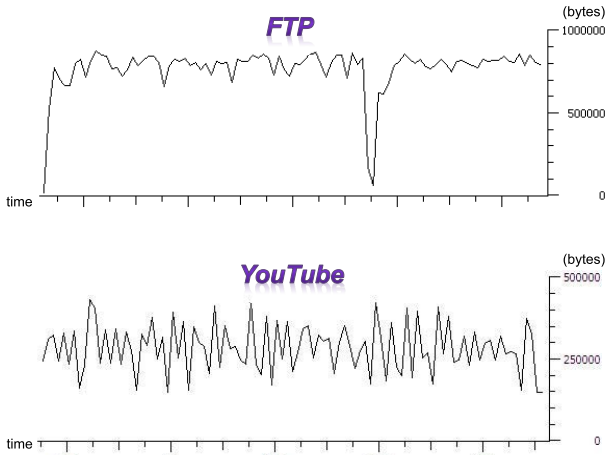


Fig. 13. Impact of lookup results on mobile communication.

AP information weighted by crowd-vehicle's reliability, the crowdsourced result (see Fig. 12(d)) shows more accurate estimations compared to the individual vehicle (see Fig. 12(b) and Fig. 12(d)). `CrowdWiFi` can look up all six Open-Mesh nodes to match with the actual locations, and the average estimation error in the real testbed experiments is 2.04m. Recall, Skyhook relies on war-driving to create a current radio map, and then updates lookup results according to its previous records. We tested Skyhook over the same area resulting in an estimation error of 11.61m; less accurate than `CrowdWiFi`.

In order to test the `CrowdWiFi` lookup results on Internet connection and data transmission, we further evaluated the Open-Mesh testbed in Fig. 10 on two types of real world traffics: FTP (for file transmission) and YouTube (for multimedia transmission). Once the vehicle obtains lookup results from its `CrowdWiFi` system, it connects to one roadside AP detected with better signal and closer location. Our evaluation results were collected when the connected AP in our experiment is the green one with number "1" included, as shown in Fig. 10. The inclusive number in node icon indicates the number of users per node (in our case here only one vehicle is connected to a mesh node so only one node icon has a number). The vehicle was moving at speed around 15mph. Fig. 13 presents the data transmission rate by the connection during a driving period around 6 seconds. The average transmission rate is around 6.4$Mbps$ for FTP traffic and 2$Mbps$ for YouTube traffic. Most of the transmission sessions are

relatively stable, in comparison with only a small portion with obvious change that was caused by roadside signal interference (*e.g.* parallel vehicles, cluster of trees, group of pedestrians). Considering the highly dynamic environments, the performance of `CrowdWiFi` presents good network connection for mobile communication in vehicular networks.

Overall, the results of these experimental evaluations confirm that our proposed approaches in `CrowdWiFi` are valid and can perform reasonably well in practice.

VII. Conclusion

`CrowdWiFi` is a system for vehicular roadside AP lookup using online CS and offline crowdsourcing techniques. The CS-based coarse-grained lookup approach includes completed online steps to make the CS operation efficient and effective for crowd-vehicles to recover sparse APs. Online CS also reduces the number of RSS readings and the size of RSS matrix required while maintaining accuracy. The crowdsourcing-based fine-grained lookup can generate truthful estimation of APs based on the efficient aggregation of sensing results using a bipartite graph, providing reliability measures of each crowd-vehicle using iterative inference. Through extensive simulation and real testbed results, we have showed the superiority of `CrowdWiFi` over existing approaches with respect to lookup errors (*e.g.* around 80% improvement on localization in comparison with the state-of-the-art Skyhook).

Appendix A

Proof of Proposition 1: From the assumption, y'_k can be written as: $y'_k = QA^{\dagger}y_k = QA^{\dagger}A\theta_k + QA^{\dagger}\varepsilon = Q\theta_k + \varepsilon'$. Since Q is usually a nearly orthogonal matrix with unit norm, it is shown in [39] that Q obeys the Restricted Isometry Property that is needed by the CS [40]. In addition, θ_k is 1-sparse and the number of RPs M is in the order of $K\log(N/K)$. Therefore, the orthogonal operations satisfy both the sparsity and incoherence requirements of CS theory [36], [39], [41], and the location indicator θ_k can be well recovered from y'_k with ℓ_1-minimization.

Appendix B

Proof of Proposition 2: Assume P_t is the transmission power of a vehicle to send one-time localization broadcast message, and P_{ri} is its reception power to receive one RSS measurement at one RP i from an AP. T_t and T_r are their time cost, respectively, which is around one second by the WiFi standard and our corresponding setting for `CrowdWiFi` operations in Section IV-A.2. Let P_l be the energy consumption for CS operation, and T_l be the time cost for its localization steps which is normally in the order of milliseconds. Then $E = P_tT_t + \sum_{i=1}^{M} P_{ri}T_r + P_lT_l$. Since $T_t \ll T_l$, $T_r \ll T_l$, and M is a small number in CS operation as explained in Section IV-A.2, we have $E \propto P_lT_l$. According to the proof on page 112 in [42], the time complexity of CS operation is $\Omega(KMN)$. We can express $T_l = \lambda N$, where $\lambda = \Omega(KM)$. Therefore, we conclude: $E \propto \lambda N, \lambda = \Omega(KM)$.

References

[1] K. Abrougui, A. Boukerche, and R. W. N. Pazzi, "Design and evaluation of context-aware and location-based service discovery protocols for vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 3, pp. 717–735, Sep. 2011.

[2] X. Wang, X. Lin, Q. Wang, and W. Luan, "Mobility increases the connectivity of wireless networks," *IEEE/ACM Trans. Netw.*, vol. 21, no. 2, pp. 440–454, Apr. 2013.

[3] U. Shevade *et al.*, "Enabling high-bandwidth vehicular content distribution," in *Proc. ACM 6th Co-NEXT*, 2010, pp. 265–276.

[4] V. Kone, H. Zheng, A. Rowstron, and B. Y. Zhao, "The impact of infostation density on vehicular data dissemination," *Mobile Netw. Appl.*, vol. 16, no. 6, pp. 807–819, 2011.

[5] OpenWrt. [Online]. Available: https://openwrt.org.

[6] DD-WRT. [Online]. Available: http://www.dd-wrt.com/site/index.

[7] FON. [Online]. Available: https://corp.fon.com/en.

[8] BT WiFi. [Online]. Available: http://www.btwifi.com.

[9] OnStar. [Online]. Available: https://www.onstar.com/us/en/home.html.

[10] T. Sukuvaara and P. Nurmi, "Wireless traffic service platform for combined vehicle-to-vehicle and vehicle-to-infrastructure communications," *IEEE Wireless Commun.*, vol. 16, no. 6, pp. 54–61, Dec. 2009.

[11] D. Wu, D. I. Arkhipov, Y. Zhang, C. H. Liu, and A. C. Regan, "Online war-driving by compressive sensing," *IEEE Trans. Mobile Comput.*, vol. 14, no. 11, pp. 2349–2362, Nov. 2015.

[12] Y. Bejerano, I. Cidon, and J. Naor, "Efficient handoff rerouting algorithms: A competitive on-line algorithmic approach," *IEEE/ACM Trans. Netw.*, vol. 10, no. 6, pp. 749–760, Dec. 2002.

[13] D. Wu, L. Bao, and C. H. Liu, "Scalable channel allocation and access scheduling for wireless Internet-of-Things," *IEEE Sensors J.*, vol. 13, no. 10, pp. 3596–3604, Oct. 2013.

[14] R. Jiang, Y. Zhu, T. He, Y. Liu, and L. M. Ni, "Exploiting trajectory-based coverage for geocast in vehicular networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 12, pp. 3177–3189, Dec. 2014.

[15] D. Wu, Y. Zhang, L. Bao, and A. C. Regan, "Location-based crowdsourcing for vehicular communication in hybrid networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 837–846, Jun. 2013.

[16] T. H. Luan, L. X. Cai, J. Chen, X. Shen, and F. Bai, "VTube: Towards the media rich city life with autonomous vehicular content distribution," in *Proc. 8th IEEE SECON*, Jun. 2011, pp. 359–367.

[17] I. Constandache, R. R. Choudhury, and I. Rhee, "Towards mobile phone localization without war-driving," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[18] A. P. Subramanian, P. Deshpande, J. Gao, and S. R. Das, "Drive-by localization of roadside WiFi networks," in *Proc. 27th IEEE INFOCOM*, Apr. 2008, pp. 718–725.

[19] Skyhook. [Online]. Available: http://www.skyhookwireless.com.

[20] G. Han, H. Xu, T. Q. Duong, J. Jiang, and T. Hara, "Localization algorithms of wireless sensor networks: A survey," *Telecommun. Syst.*, vol. 52, no. 4, pp. 2419–2436, 2013.

[21] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[22] C. Feng, W. Au, S. Valaee, and Z. Tan, "Compressive sensing based positioning using RSS of WLAN access points," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

[23] B. Zhang, X. Cheng, N. Zhang, Y. Cui, Y. Li, and Q. Liang, "Sparse target counting and localization in sensor networks based on compressive sensing," in *Proc. IEEE INFOCOM*, Apr. 2011, pp. 2255–2263.

[24] N. M. Do, C.-H. Hsu, and N. Venkatasubramanian, "CrowdMAC: A crowdsourcing system for mobile access," in *Proc. ACM/IFIP/USENIX Middleware*, 2012, pp. 1–20.

[25] J. Zhao *et al.*, "Localization of wireless sensor networks in the wild: Pursuit of ranging quality," *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 311–323, Feb. 2013.

[26] Y. Zhang, L. Bao, S.-H. Yang, M. Welling, and D. Wu, "Localization algorithms for wireless sensor retrieval," *Comput. J.*, vol. 53, no. 10, pp. 1594–1605, 2010.

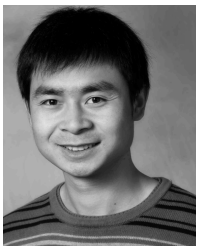
[27] J. Koo and H. Cha, "Autonomous construction of a WiFi access point map using multidimensional scaling," in *Proc. 9th ACM Pervasive*, 2011, pp. 115–132.

[28] Y.-C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm, "Accuracy characterization for metropolitan-scale Wi-Fi localization," in *Proc. ACM MobiSys*, 2005, pp. 233–245.

[29] Z. Yang, C. Wu, and Y. Liu, "Locating in fingerprint space: Wireless indoor localization with little human intervention," in *Proc. ACM Mobicom*, 2012, pp. 269–280.

[30] D. Wu, L. Bao, and R. Li, "Robust localization protocols and algorithms in wireless sensor networks using UWB," *Ad Hoc Sensor Wireless Netw.*, vol. 11, nos. 3–4, pp. 219–243, 2011.
[31] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-effort crowdsourcing for indoor localization," in *Proc. ACM Mobicom*, 2012, pp. 293–304.
[32] S. Yang, P. Dessai, M. Verma, and M. Gerla, "FreeLoc: Calibration-free crowdsourced indoor localization," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2481–2489.
[33] J. Zhu, K. Zeng, K.-H. Kim, and P. Mohapatra, "Improving crowdsourced Wi-Fi localization systems using bluetooth beacons," in *Proc. IEEE 9th SECON*, Jun. 2012, pp. 290–298.
[34] V. C. Raykar *et al.*, "Learning from crowds," *J. Mach. Learn. Res.*, vol. 11, pp. 1297–1322, 2010.
[35] D. R. Karger, S. Oh, and D. Shah, "Iterative learning for reliable crowdsourcing systems," in *Proc. Neural Inf. Process. Syst. (NIPS)*, 2011, pp. 1953–1961.
[36] E. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Problems*, vol. 23, no. 3, pp. 969–985, 2007.
[37] *NCTUns 5.0, Network Simulator and Emulator*. [Online]. Available: http://NSL.csie.nctu.edu.tw/nctuns.html.
[38] Open-Mesh. [Online]. Available: http://www.open-mesh.com.
[39] E. J. Candès and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inf. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.
[40] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin, "A simple proof of the restricted isometry property for random matrices," *Constructive Approx.*, vol. 28, no. 3, pp. 253–263, Dec. 2008.
[41] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207–1223, 2006.
[42] C. Feng, "Research and implementation of a compressive sensing-based indoor positioning system using RSS," Ph.D. dissertation, Dept. Commun. Inf. Syst., Beijing Jiaotong Univ., Beijing, China, 2010.

Di Wu received the M.S. and Ph.D. degrees in computer science from the University of California at Irvine, USA, in 2012 and 2013, respectively. He was a Staff Research Associate with the University of California at Irvine, a Visiting Researcher with IBM Research, and a Research Associate with the Stanford Research Institute. He is currently an Associate Professor with the Department of Computer Engineering, Hunan University, Changsha, China. He is also a Research Associate with Imperial College London, the Intel Collaborative Research Institute for Sustainable Connected Cites, and the University of California Transportation Center. His research interests include wireless networks and mobile computing, Internet-of-Things and cyberphysical systems, smart cities, and big data. He has actively served on many conference committees, and is an Associate Editor of the IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. He is a member of the IEEE and the ACM.



Qiang Liu received the B.S. degree in information and computation science from Beihang University, Beijing, China, in 2008, and the M.S. degree in mathematical, computational and systems biology, and the Ph.D. degree in computer science from the University of California at Irvine, USA, in 2009 and 2014, respectively. He was a Research Intern with Microsoft Research Redmond, and a Post-Doctoral Researcher with the MIT Computer Science and Artificial Intelligence Laboratory. He is currently an Assistant Professor with the Department of Computer Science, Dartmouth College. His research area is machine learning and statistics, with interests spreading over the pipeline of data collection (mainly crowdsourcing), learning, inference, decision making, and various applications under the framework of probabilistic graphical models.

Yong Li received the B.S. degree in electronics and information engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2007, and the Ph.D. degree in electronics engineering from Tsinghua University, Beijing, China, in 2012. He was a Visiting Research Associate with Telekom Innovation Laboratories and The Hong Kong University of Science and Technology, and a Visiting Scientist with the University of Miami, FL, USA. He is currently an Assistant Professor with the Department of the Electronic Engineering, Tsinghua University. His research interests include mobile opportunistic networks, device-to-device communication, software-defined networks, network virtualization, and future Internet. He is a member of the IEEE and the ACM.

Julie A. McCann is currently a Professor of Computer Systems with Imperial College London. She leads both the Adaptive Embedded Systems Engineering Research Group and the Intel Collaborative Research Institute for Sustainable Cities, and is with NEC and others on substantive smart city projects. Her research centers on highly decentralized and self-organizing scalable algorithms for spatial computing systems, e.g., wireless sensing networks. She is a member of the IEEE and the ACM as well as a Chartered Engineer, and was elected as a fellow of BCS in 2013. She has received significant funding through bodies, such as the U.K.'s EPSRC, TSB, and NERC, and various international funds, and is an Elected Peer Member of EPSRC. She has actively served on, and chaired, many conference committees, and is an Associate Editor of the *ACM Transactions on Autonomous and Adaptive Systems*.

Amelia C. Regan received the B.A.S. degree in systems engineering from the University of Pennsylvania, the M.S. degree in applied mathematics from Johns Hopkins University, and the M.S. and Ph.D. degrees in transportation systems engineering from the University of Texas at Austin. She has taught short courses with the Athens University of Business and Economics and the National Technical University of Denmark, and was an Operations Research Analyst with the Association of American Railroads and United Parcel Service, prior to receiving the Ph.D. degree. She is currently a Professor of Computer Science and Transportation Systems Engineering with the University of California at Irvine. Her research is focused on algorithm development for optimization of transportation and communication systems.

Nalini Venkatasubramanian received the M.S. and Ph.D. degrees in computer science from the University of Illinois at Urbana–Champaign. She was a member of the Technical Staff with Hewlett-Packard Laboratories, Palo Alto, CA, for several years, where she worked on large-scale distributed systems and interactive multimedia applications. She has worked on various database management systems and on programming languages/compilers for high performance machines. She is currently a Professor with the School of Information and Computer Science, University of California at Irvine. Her research interests include distributed and parallel systems, middleware, mobile environments, multimedia systems/applications, and formal reasoning of distributed systems. She is a Senior Member of the IEEE and the ACM.