Molé: a Scalable, User-Generated WiFi Positioning Engine

Jonathan Ledlie*, Jun-geun Park**, Dorothy Curtis**, André Cavalcante***, Leonardo Camara***, and Robson Vieira***
*Nokia Research Center, Cambridge, USA. Email: jonathan.ledlie@nokia.com
**Massachusetts Institute of Technology, Cambridge, USA. Email: {jgpark,dcurtis}@mit.edu
***Nokia Institute of Technology, Manaus, Brasil. Email: {andre.cavalcante,leonardo.camara,robson.domingos}@indt.org.br

***Abstract*—We describe the design, implementation, and evaluation of Molé, a mobile organic localization engine. Unlike previous work on crowd-sourced WiFi positioning, Molé associates a hierarchical name with each place. By not relying on a map and by being more strict than uninterpreted names for places, Molé aims for a more flexible and scalable point in the design space of localization systems. Molé also employs a new statistical positioning algorithm to differentiate between neighboring places. This algorithm accounts for temporal variations in a place's fingerprint in a principled manner, allows for aggregation of fingerprints from many users, and is compact enough for on-device storage.**

***Index Terms*—crowd-sourcing, WiFi positioning.**

I. Introduction

The ability for a mobile device to perceive a user's location has many applications, from social networking "check-ins" to location-appropriate content, such as automatically presenting people with a relevant train schedule.

While the global positioning system (GPS) enables devices to sense their location in most outdoor environments, bad weather and "urban canyons" can restrict its operation. In addition, there are many indoor positioning applications where GPS can provide only limited assistance, as it typically provides a position fix only near windows and doors.

To enable room-grain indoor and outdoor positioning in GPS-less environments, researchers have used physically-fixed wireless beacons to associate a unique "fingerprint" with each place or grid point [1]–[4]. While the types of wireless beacons have varied over time, most techniques now use 802.11 WiFi beacons because of their near ubiquity, particularly in urban and suburban environments. Because of the difficulty in translating between distance and received signal strength [5], more compact alternatives to fingerprinting – e.g., triangulating among the beacons – are generally eschewed.

One of the key problems with fingerprinting, however, is learning the fingerprint for each place – however "places" are designated. We call the process where a person links a fingerprint to a place "binding." Several commercial vendors offer positioning services, which include the fingerprint-generation survey [6]. However, these come at a steep price: a large office building can cost $10,000 USD with no maintenance included. Because this is prohibitively expensive for many applications – such as contextualizing a device's behavior based on which room of a house it is in – several research groups have begun to crowd-source fingerprints from end-users [7]–[10]. In the model for these Wikipedia-style approaches, a single locally-knowledgeable user performs the bind for a place and many visitors can then rely on the database of fingerprints there.

Molé focuses on a new point in the design space in crowd-sourced, or "organic," positioning systems. Some systems, such as OIL [8], present a map to the user: users bind places by clicking on the map. Others, like Redpin [7], allow the association of any text string with a place's fingerprint. In contrast, Molé arranges the world hierarchically; this imposes a clean, intuitive namespace (country, region, ...), and allows for data prefetching at a building scale if not larger. It also isolates problems in the fingerprint database to only portions of the tree. While Molé could use any localization algorithm, it relies on a new compact data structure that allows many fingerprints to be stored on the user's device. In turn, this allows the user's device – not a server – to differentiate among potential places with similar fingerprints, improving privacy.

II. Model of Places

Molé arranges the discrete, human-designated places of the world in a hierarchy. While the hierarchy could be of variable depth, our current implementation contains five levels, as the estimate in Figure 1 illustrates. From coarse to fine, the levels typically refer to country, region, city, area, and unique place. Areas are the unit of fingerprint aggregation, transfer, and, therefore, privacy; the server can know at most what areas you visit. Areas typically refer to street addresses (e.g., "4 Cambridge Center" in Figure 1), although they could refer to larger outdoor areas such as parks. The design would also allow aggregation at higher levels.

We believe that arranging places in a hierarchy is useful in many organic positioning settings. Earlier approaches have used visual maps [8], [9] or uninterpreted strings [7] to identify individual places. Visual maps require that a fairly accurate map of the area – typically a building – exists. While well-managed places, such as universities and airports, may be able to generate maps, this approach may not scale to individual homes or businesses, where people may not have the time, knowledge, or interest to create a map of their sets of places. In addition, many users find it non-trivial to locate themselves on indoor maps, particularly in complex buildings. Assigning uninterpreted strings to places during a bind has its own

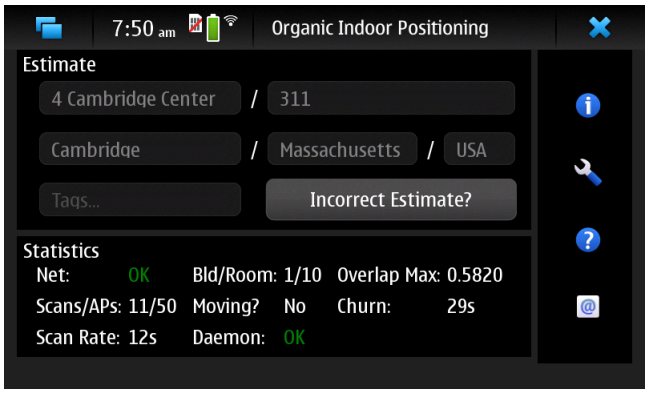

Fig. 1. **Molé User Interface.** It shows the country, region, city, area, room hierarchy in street address format.

challenges: for example, the namespace may rapidly become crowded with similar names. While Barry et al. do allow for spaces within buildings [10], their hierarchy is not intended to cover the world.

Figure 1 shows the current estimate of a device's position within this hierarchy. Users click on the "Incorrect Estimate?" button to edit the current estimate and make a new bind, improving future estimates for themselves and other users. The statistics are explained in Section IV.

III. Algorithm

Our localization algorithm is a new, statistical variation on the standard RF fingerprint method [1], [3]. It is called *Ma*ximum weighted *O*verlap, or MAO. The two key advantages to MAO are (1) that it is extremely simple to compute and (2) that it provides a general scan distance function, which can be used to estimate physical distances between sets of fingerprinted objects. Because we anticipate localization algorithms running continuously in the background on mobile devices, this simple computation should translate into longer battery life. Scan distance functions are also useful for clustering scans. Clustering, in turn, can be used for outlier detection and cleaning scan databases. By themselves, distance functions are also useful for estimating the physical distance between the positions where the scans were made.

To create a MAO fingerprint, we begin with a standard set of place-to-APs histograms containing raw RSSI readings [3]. As in [3], we summarize each per-place per-AP histogram with a single Gaussian with mean μ and standard deviation σ. Every place is assigned a fingerprint, which is a set of mappings from access points to data triples:

$$AP_i \Rightarrow \langle w_i, \mu_i, \sigma_i \rangle \tag{1}$$

where w_i is the weight of AP_i, the number of observable APs is τ, and the total weight for each fingerprint $\sum_{i=1}^{\tau} w_i$ is 1. Note that the most recent k scans of the user also form a fingerprint using the same method.

Determining the weight w to apply to each visible AP is an important component of our algorithm. A strawman method would be to simply weight each visible AP equally: $1/\tau$.

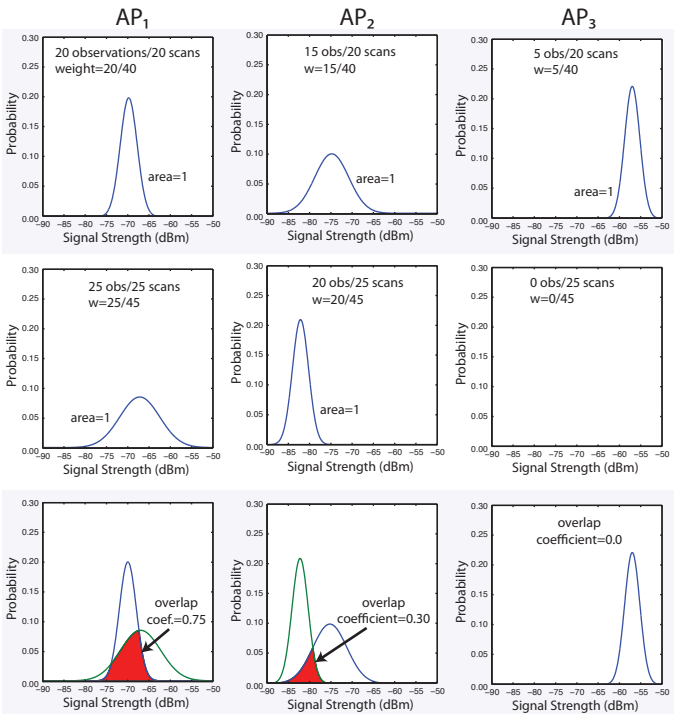

Fig. 2. Example of MAO: $place_1$ (top) $\cup$ $place_2$ (middle) = Overlap (bottom). The 20 scans in $place_1$ have observed three different access points: AP_1, AP_2, AP_3. The 25 scans in $place_2$ have observed two different access points, two of which are the same as those seen in $place_1$: AP_1, AP_2. AP_3 was not observed in $place_2$. To compute the weights for $place_1$, we divide the observations for each AP by the total number of observations: $20 + 15 + 5 = 40$. The same procedure is done for $place_2$. This completes the creation of fingerprints for these two places. The bottom row shows how the distance between the two fingerprints for places 1 and 2 is computed: $s = 0.75 \times \frac{20/40+25/45}{2} + 0.30 \times \frac{15/40+20/45}{2} - 5/40$. "$Place_2$" could equivalently be a set of scans as seen by an end-user's device: the algorithm to compute their overlap score would be the same.

Instead, we base the weight on the probability that the given AP will actually be observed in the place. When a place is scanned many times, some APs will be seen in every scan, and some seen only rarely. This captures the intuition that a user's device will see the same APs with the same signal strength distribution *and* the same observation frequency when it is in the same place (these two quantities are only weakly correlated). If the user's fingerprint does not contain an AP that is almost always observed when in a particular place, it is highly unlikely that the user is in this place. Weighting according to scan detection frequency reflects this intuition. Specifically, the weight is:

$$w_i = r_i / \sum_{i=1}^{\tau} r_i \tag{2}$$

where r_i is the number of readings of AP_i.

To find the distance between two fingerprints, we determine the similarity in signal strengths of APs that exist in both fingerprints, and penalize for missing APs. The comparison of any two fingerprints returns a distance $-1 \leq D \leq 1$, where a comparison of identical fingerprints returns 1 and of disjoint

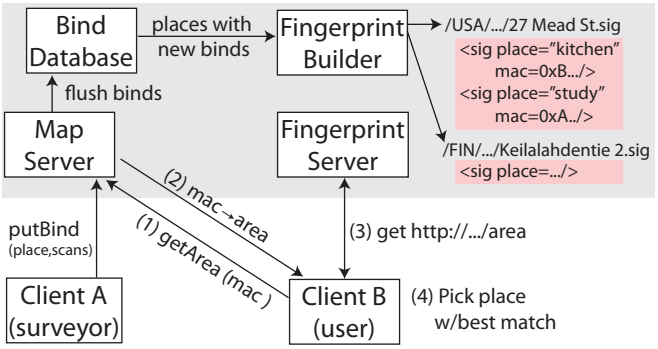

Fig. 3. Molé's server-side components.

fingerprints returns -1 (Disjoint fingerprints are those that share no access points). For fingerprints A and B, let $D(A,B) = \sum_{i \in A \cup B} \delta_i$, where the effect each AP_i is:

$$\delta_i = \begin{cases} \frac{\omega_a + \omega_b}{2} \times O\left(\mu_a, \sigma_a, \mu_b, \sigma_b\right) & \text{if } i \in A, i \in B, \\ \frac{-\omega_a}{2} & \text{if } i \in A, i \notin B, \\ \frac{-\omega_b}{2} & \text{if } i \notin A, i \in B, \end{cases} \quad (3)$$

where $O()$ is the overlap coefficient between the two Gaussian distributions [11]. Figure 2 provides an example of computing the distance between a pair of fingerprints.

One particularly nice aspect of this overlap computation is that it exists as a closed-form function [11]. Alternatively, the results from the function can be stored in a look-up table [12]; we found a table with only hundreds of values gave almost the same results as a function. This simple computation is in contrast to graphical models [13], which can require thousands of iterations to converge to a place.

MAO admits a few variations. First, the selection of a Gaussian function is orthogonal to the method. While RSSI distributions often do follow a Gaussian and often can be summarized with one, any density function that fits the data would work. In particular, places could have their own individual functions (e.g., Beta distribution), which do occasionally fit RSSI data better. The downside to selecting a different function is that the overlap computation might be more computationally expensive. Second, variations on the weighting function are also possible, e.g., not dividing by two when an AP is missing. This would serve to penalize missing APs more (and increase the domain of s to -2). Alternatively, applying no penalty may improve accuracy in many situations.

Details such as dealing with only one observation have been omitted due to space constraints.

IV. Implementation

Molé's implementation is divided into client and server components. The client portion periodically scans WiFi signals and makes an estimate of the current place available to other applications on the same mobile device.

The client itself is made up of two parts: a daemon, which runs continuously in the background, and a user interface, which is displayed when the user wants to make a bind, modify the daemon's behavior, or view statistics. Figure 1 shows the user interface. Its statistics include: the number of scans being used to form the estimate; the count of distinct APs that were observed within these scans; the current scan period; the number of areas and individual places within those areas under MAO consideration; whether the user is moving; the score of the current estimate ("overlap max"); and churn, the time since the estimate was last changed.

As Haeberlen et al. showed [3], comparing more user scans against each fingerprint improves spot-on accuracy, with diminishing returns after about eight scans with their data. But more frequent scanning reduces battery life, and having a fixed, large number of user scans introduces a lag when the user is moving between places. If a device has an accelerometer, Molé uses it to attempt to find a happy medium between battery consumption and update lag. If the device is deemed stationary, it slows down the scan rate and other functions. When walking is detected, the current set of user scans is discarded and the scan rate is increased (up to once per 10s in our current implementation). By truncating the user scans (11 in Figure 1), Molé returns a less accurate, but more timely estimate. When the user stops moving, the user scans accumulate and the estimate improves. Because we simply truncate the localization and bind queue in response to movement, our method is independent of the choice of the particular motion detection algorithm; we use Shafer and Chang's detector [14]. An alternative method would treat the detector as less of a black box and could dynamically adjust the length of the queues based on the magnitude or confidence with which movement was detected.

Figure 3 shows Molé's four main server components and the key methods clients use to make binds and access fingerprints. Molé's server side is designed to run elastically "in the cloud:" its client-facing components, the Map Server and Fingerprint Server are easy to replicate. The figure shows the two paths of client actions: (a) binding and (b) localizing. A client bind is sent to the Map Server, which acts as a write-back cache. The Fingerprint Builder periodically monitors the database for places with new binds (or entirely new places). For each of these places, it aggregates all recent binds and determines a new fingerprint. Fingerprints for other places in the same area are cached in the database. The builder then writes out each changed area's collection of fingerprints in a single area fingerprint file. Because these files change infrequently and are named by the fully-qualified area, they can be trivially cached, versioned, and diffed.

Client localization involves accessing the correct area's fingerprint file (if it is not cached on-device), filtering down to a few fingerprints, and producing a top estimate with MAO. The client periodically asks the server for the list of areas associated with one of its visible MACs (step 1), and receives the fully-qualified (hierarchical) area name in response (step 2). It then requests the area's fingerprint file (step 3) and localizes using the current user scans (step 4). To reduce the number of fingerprints that MAO must compare, we apply Charrow's fingerprint filtering to our local cache to identify a set of "nearby" locations [15]; in Figure 1, ten places have

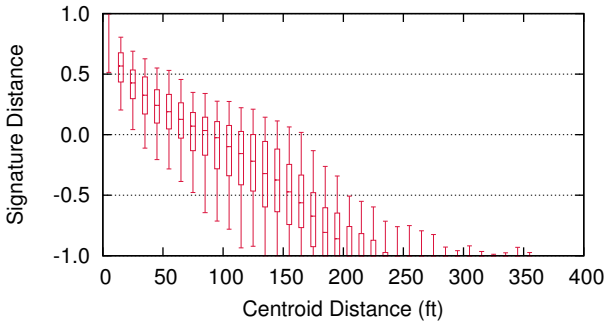

Fig. 4. Signal-to-Physical Distance Correlation. The data show that there is a strong correlation between fingerprint distance and physical distance, particularly when spaces are nearby to one another. Space distance is measured as centroid-to-centroid in a nine-story building which contains more than 1,400 distinct spaces. The top and bottom whiskers show the top 5% and bottom 5%, respectively. Boxplot lines mark the 25^{th}, 50^{th}, and 75^{th} pcts.

passed this filter.

The source code for Molé has been released under an open source license and we invite contributions[1].

V. Evaluation

We have successfully tested Molé in preliminary trials at several labs. Due to space constraints, we summarize the results we include in the full version of this paper:

- We find that MAO has favorable accuracy results as compared to a state-of-the-art Bayes localizer, often achieving better performance when places are physically adjacent.
- We find that use of a movement detector can result in a dramatic improvement in update delay and in unpolluted signature creation.
- We show how MAO can be used to estimate the physical distance between two objects as shown in Figure 4.
- Through a deployment in a three story building, we show that Molé can rapidly crowd-source an accurate location system.

VI. Related Work

While there is much previous research on indoor positioning in general [1]–[4], [7]–[10], [16], [17], for this short paper, we only differentiate between MAO and previous work. The closest related work to the MAO localization method is a paper by Lemelson et al. [18]. In this work, the authors use basic Gaussian overlap – without any weighting – to determine the similarity of the fingerprint of one point with adjacent points (in contrast to our per-space fingerprints, this research group prefers to associate fingerprints with points on an artificial grid which is overlaid on the space map). Lemelson et al. do not use the overlap method for localization at all. Instead, they use it to determine the similarity of one point's Gaussian fingerprint with another. They are attempting to anticipate the likely estimate localization error for a given point. They show that points with very similar fingerprints (as determined by the overlap function) tend to have poor localization accuracy, because they are often confused with adjacent points. Our results (not included here) show that, without weighting, basic Gaussian overlap is clearly inferior to Gaussian overlap with weighting based on the frequency of AP reception. An example where this method without the weighting would clearly perform poorly is a case where many spaces occasionally hear from many APs, and always hear from exactly one unique AP. In this case, the no-weighting method would lose the single unique characteristic in the noise, whereas the weighting method would select the right space. Although this is an extreme example, it illustrates a problem that often occurs in environments with dense but intermittent AP coverage.

[1]http://github.com/organic-positioning

VII. Conclusion

This paper presented Molé, a mobile organic localization engine, specifically designed for large-scale positioning. We focused on three key components of Molé: its hierarchical arrangement of places, its simple localization algorithm, and its "cloud"-based server design. Together, these components contribute to a positioning system that can run compactly on a broad range of mobile devices and scale worldwide. For the cases where visual maps do exist, we plan on extending the hierarchical and scalable structure of Molé to a visual map-based UI.

References

[1] P. Bahl *et al.*, "RADAR: An In-Building RF-Based User Location and Tracking System," in *INFOCOM*, Tel Aviv, Israel, Mar. 2000.
[2] N. Priyantha, A. Chakraborty, and H. Balakrishnan, "The Cricket Location-Support System," in *MOBICOM*, 2000.
[3] A. Haeberlen, E. Flannery *et al.*, "Practical Robust Localization over Large-Scale 802.11 Wireless Networks," in *MOBICOM*, 2004.
[4] J. Krumm and J. Platt, "Minimizing Calibration Effort for an Indoor 802.11 Device Location Measurement System," Microsoft Research, Tech. Rep. MSR-TR-03-82, Nov. 2003.
[5] K. Pahlavan *et al.*, "Wideband Radio Propagation Modeling for Indoor Geolocation Applications," *IEEE Comm. Mag.*, April 1998.
[6] "Ekahau positioning engine," ekahau.com.
[7] P. Bolliger, "RedPin: Adaptive, Zero-Configuration Indoor Localization," in *LoCA*, Oberpfaffenhofen, Germany, 2008.
[8] J. Park *et al.*, "Growing an Organic Indoor Location System," in *MobiSys*, San Francisco, CA, Jun. 2010.
[9] E. S. Bhasker, S. W. Brown *et al.*, "Employing user feedback for fast, accurate, low-maintenance geolocationing," in *PerCom*, 2004.
[10] A. Barry, B. Fischer, and M. Chang, "A long-duration study of user-trained 802.11 localization," in *MELT*, Orlando, FL, Sep. 2009.
[11] H. F. Inman and E. L. Bradley, "The overlapping coefficient as a measure of agreement between probability distributions and point estimation of the overlap of two normal densities," *Communications in Statistics-Theory and Methods*, vol. 18, no. 10, pp. 3851–3874, 1989.
[12] J. M. Linacre, "Overlapping Normal Distributions," *Rasch Measurement Transactions*, vol. 10, no. 1, 1996.
[13] D. Madigan, E. Einahrawy *et al.*, "Bayesian Indoor Positioning Systems," in *INFOCOM*, Miami, FL, March 2005.
[14] I. Shafer and M. L. Chang, "Movement Detection for Power-Efficient Smartphone WLAN Localization," in *WSWiM*, Miami Beach, FL, 2010.
[15] B. Charrow, "Organic indoor location: Infrastructure and applications," Master's thesis, Massachusetts Institute of Technology, Feb. 2010.
[16] T. Gallagher, B. Li, A. G. Dempster, and C. Rizos, "A sector-based campus-wide indoor positioning system," in *IPIN*, Zurich, 2010.
[17] A. LaMarca *et al.*, "Place Lab: Device Positioning Using Radio Beacons in the Wild," in *Pervasive*, Munich, Germany, May 2005.
[18] H. Lemelson, M. Kjaergaard *et al.*, "Error Estimation for Indoor 802.11 Location Fingerprinting," in *LoCa*, Tokyo Japan, May 2009.