Chameleon: Survey-Free Updating of a Fingerprint Database for Indoor Localization

In fingerprint-based indoor localization, keeping the fingerprint current is important for localization accuracy. Chameleon is a novel survey-free approach to localize users and maintain an updated fingerprint database despite fingerprint signal changes. Extensive experimental trials confirm Chameleon's potential.

Suining He, Bo Ji, and S.-H. Gary Chan
The Hong Kong University of Science and Technology

Location-based services (LBS) have been attracting attention in recent years. Despite much commercial success for outdoor LBS using GPS, indoor localization remains challenging due to the absence of GPS signals.[1]

With the ubiquitous deployment of wireless local area networks (WLANs), localization systems that use Wi-Fi signals as a "fingerprint" have recently attracted attention.[2] The fingerprinting approach comprises two phases. In the *survey phase*, a site survey is conducted at different reference points (RPs) of known positions to create a fingerprint database. In the *query phase*, a mobile agent (user) collects signal measurements at its location. With these measurements, the server then estimates the agent's location.[2–4]

Clearly, localization accuracy hinges on how well the fingerprint database reflects the current signal environment. This environment is by no means static due to environmental changes, such as the introduction or removal of access points (APs), the establishment or tear-down of partitions, or the adjustment of AP power (because of coverage extension, interference avoidance, wearing, and so on). Usually, the number of such *altered APs* is few compared to the total APs at a location (say 1–4 alterations out of approximately 10–20 signals collected). As AP signals change, the actual fingerprint then differs from the one previously collected at the same location and stored in the database. Long-term altered AP signals—those altered over days or even months—have a far larger impact on location accuracy than short-term transient signal fluctuations (in seconds or minutes).

Figure 1 illustrates a heat map of two altered APs during two site surveys conducted in our campus atrium (2,000 m^2) on 10 October and 10 December 2014. During each site survey, we collected 520 RPs at 5 m grid density (with 28 physical APs detected per RP on average). As the figure shows, the transmission power and installed locations of the APs changed due to an academic building renovation. If these altered APs aren't known during location estimation, it can result in high localization errors.

To address this issue, we propose Chameleon, which offers novel and highly accurate fingerprinting localization when APs might have been altered. (If the environment experiences drastic change, a full site survey should be conducted.)

Published by the IEEE CS ■ 1536-1268/16/$33.00 © 2016 IEEE

Related Work on Survey Reduction and Fingerprint Updates

To keep the radio maps up to date, the indoor site where a localization system is deployed has traditionally been surveyed regularly[1,2] and frequently to collect new signal signatures (fingerprints). In this *surveyor approach*, trained surveyors walk through the entire site to collect fingerprints and upload them and the associated locations to the database, which is laborious and time-consuming.

Some *model-based approaches* use signal propagation models and Wi-Fi monitors[3-6] to predict the spatial distribution changes of signals and update the database. Because additional, specialized infrastructures are usually needed, these approaches might not be cost-effective for dense deployment in large indoor sites (such as airports).

Recently, a *crowdsourcing approach* has been studied, where the fingerprint database is updated in an "organic" participatory manner,[7] relying on the feedback of users to record new fingerprint data at certain locations.[7,8] Although this approach greatly reduces the survey cost, it requires explicit user participation and thus can be intrusive or inconvenient for users. It's also difficult to expect normal users to have prior knowledge of the indoor site for collecting fingerprints.

More recent approaches try to involve unconscious users in collecting fingerprints. WILL[9] and Zee[10] use dead reckoning in mobile devices to predict locations, associating the fingerprints collected by users with these locations. However, they require accurate motion sensor calibration and a known starting location for users.

Pattern-based approaches, such as UnLoc,[11] EZ,[12] and Walkie-Markie[13] propose using the Wi-Fi landmarks and signal propagation to constrain the location estimation. They don't rely on dense site-survey data and thus reduce labor-intensive efforts. However, they work best in a narrow indoor space, where the signal patterns are distinguishable.

REFERENCES

1. P. Bahl and V.N. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proc. 19th Ann. Joint Conf. IEEE Computer and Comm. Societies* (Infocom), 2000, pp. 775–784.
2. M. Youssef and A. Agrawala, "The Horus Location Determination System," *Wireless Networks*, vol. 14, no. 3, 2008, pp. 357–374.
3. Y. Gwon and R. Jain, "Error Characteristics and Calibration-Free Techniques for Wireless LAN-Based Location Estimation," *Proc. 2nd Int'l Workshop on Mobility Management & Wireless Access Protocols* (MobiWac), 2004, pp. 2–9.
4. Y. Ji et al., "ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System," *Proc. 4th Int'l Conf. Mobile Systems, Applications and Services* (MobiSys), 2006, pp. 151–164.
5. H. Lim et al., "Zero-Configuration Indoor Localization over IEEE 802.11 Wireless Infrastructure," *Wireless Networks*, vol. 16, no. 2, 2010, pp. 405–420.
6. C. Xu et al., "Towards Robust Device-Free Passive Localization through Automatic Camera-Assisted Recalibration," *Proc.10th ACM Conf. Embedded Network Sensor Systems* (SenSys), 2012, pp. 339–340.
7. J.-G. Park et al., "Growing an Organic Indoor Location System," *Proc. 8th Int'l Conf. Mobile Systems, Applications, and Services* (MobiSys), 2010, pp. 271–284.
8. O. Kaltiokallio, M. Bocca, and N. Patwari, "Follow @Grandma: Long-Term Device-Free Localization for Residential Monitoring," *Proc. IEEE 37th Conf. Local Computer Networks Workshops* (LCN Workshops), 2012, pp. 991–998.
9. C. Wu et al., "WILL: Wireless Indoor Localization without Site Survey," *IEEE Trans. Parallel and Distributed Systems*, vol. 24, no. 4, 2013, pp. 839–848.
10. A. Rai et al., "Zee: Zero-Effort Crowdsourcing for Indoor Localization," *Proc. 18th Ann. Int'l Conf. Mobile Computing and Networking* (MobiCom), 2012, pp. 293–304.
11. H. Wang et al., "No Need to War-Drive: Unsupervised Indoor Localization," *Proc. 18th Ann. Int'l Conf. Mobile Computing and Networking* (MobiCom), 2012, pp. 197–210.
12. K. Chintalapudi, A. Padmanabha Iyer, and V.N. Padmanabhan, "Indoor Localization without the Pain," *Proc. 16th Ann. Int'l Conf. Mobile Computing and Networking* (MobiCom) 2010, pp. 173–184.
13. G. Shen et al., "Walkie-Markie: Indoor Pathway Mapping Made Easy," *Proc. 10th USENIX Conf. Networked Systems Design and Implementation* (USENIX NSDI), 2013, pp. 85–98.

Chameleon is transparent to users, employing implicit crowdsourcing to achieve survey-free updating of its database to reflect the current AP signals. The basic idea is that users in the indoor space play the roles of both location "queriers" and "surveyors" at the same time (without their awareness). As their measured RSS samples capture the current indoor signal characteristic, we use the samples to localize users and update the radio map in a timely manner.

A Novel Approach

Developing techniques that can reduce survey costs and update fingerprints has long been of interest to researchers (see the "Related Work on Survey Reduction and Fingerprint Update" sidebar). In contrast to these works, Chameleon has several strengths. It doesn't need explicit user participation or user location inputs to collect current fingerprints. Furthermore, it doesn't require any costly expert surveyors. We use only the measured RSS data of (naive) clients

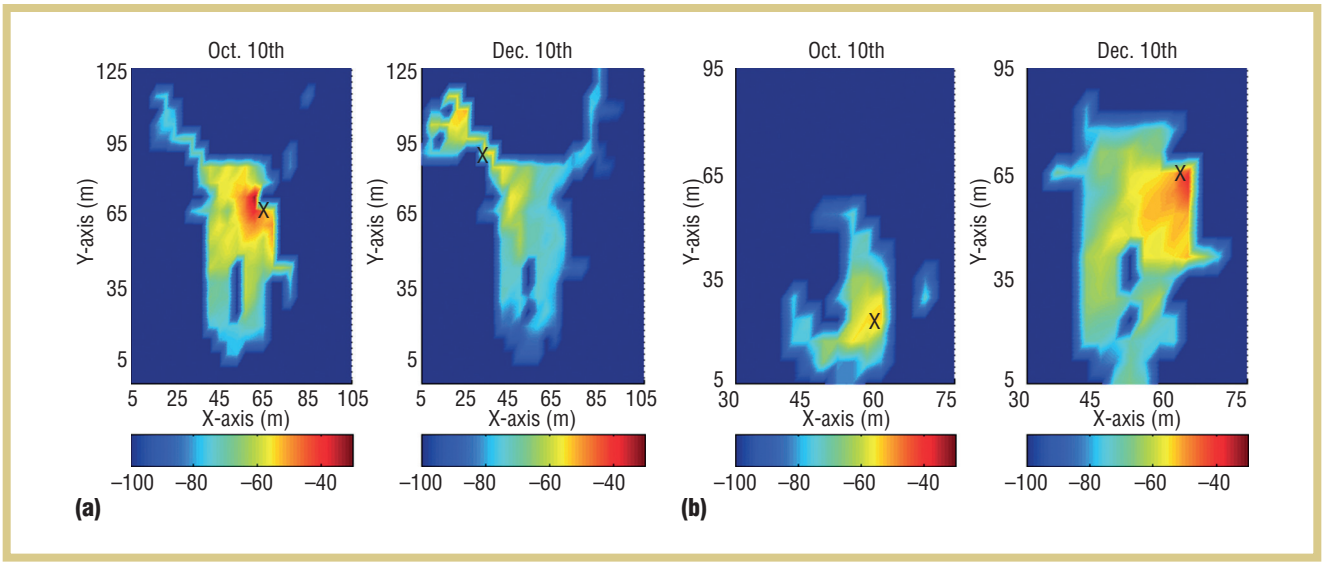


Figure 1. Change of AP signals in The Hong Kong University of Science and Technology (HKUST) atrium (black crosses indicate the potential locations of two Wi-Fi APs): (a) media access control (MAC) address 19-BE-00-05-BO and (b) MAC address 23-EB-3A-OC-35.

to update the stored radio map to the current signal environment.

On the surface, this approach seems counter-intuitive. After all, how can a client, without knowing his or her own location, be an unconscious surveyor at the same time, in the presence of possibly altered APs? To address this, Chameleon must answer two critical questions using only the measured RSS data of its clients:

- How do you estimate indoor location by identifying and filtering out altered APs?
- How do you update the database continuously to adapt to environmental signal changes, reflecting the current fingerprint in a timely manner?

Chameleon is based on the insight that any arbitrary sizable subset of unaltered APs can be used for accurate localization. Chameleon therefore filters out altered APs with an efficient clustering algorithm. Chameleon is independent of and amenable to any localization techniques. After the altered APs are filtered, any existing algorithm can localize the user. With the user location, a database update algorithm is used to adapt the radio map to the collected RSS data. Although Chameleon is studied in the context of Wi-Fi signals, it can be applied to any fingerprint-based localization system, such as FM and channel state information.

We conducted an extensive simulation and experimental study on our campus and at an international airport to evaluate Chameleon's performance.

System Overview

Chameleon is based on the following observations:

- Locations estimated from subsets of unaltered APs tend to be similar and thus cluster together, because these signal subsets are consistent with the fingerprints in the database, which leads to high localization accuracy.
- Locations estimated from subsets consisting of altered APs tend to be dispersed in nature, because they're inconsistent with the stored fingerprints, and localization error is high in these cases.

Figure 2a shows a typical example of the locations estimated from random subsets of RSS based on our experiments on our campus. We randomly corrupt two of the APs by reducing their power and collect RSS at a known location (detailed experimental settings will be described later). With the 25 APs collected, we form random subsets, and with each subset, we estimated the position and represented it as a cross or circle.

It's clear from Figure 2a that the locations estimated with the subsets formed by unaltered APs are consistent and cluster around the true user location, while those estimated by the subsets consisting of altered APs are scattered and dispersed. We thus can apply a clustering algorithm. User location is at the "dense" cluster. Our approach is similar in spirit to the RANdom SAmpling Consensus (RANSAC)[5] in identifying and filtering altered APs. Using it, Chameleon can transparently adapt the fingerprints and accurately localize users despite AP signal alterations.

Figure 2b shows the overall workflow process of Chameleon, which comprises the following three phases.

Fast Fingerprint-Integrity Detection

Normally, APs are altered only occasionally, and such alteration wouldn't affect large geographical scope. The

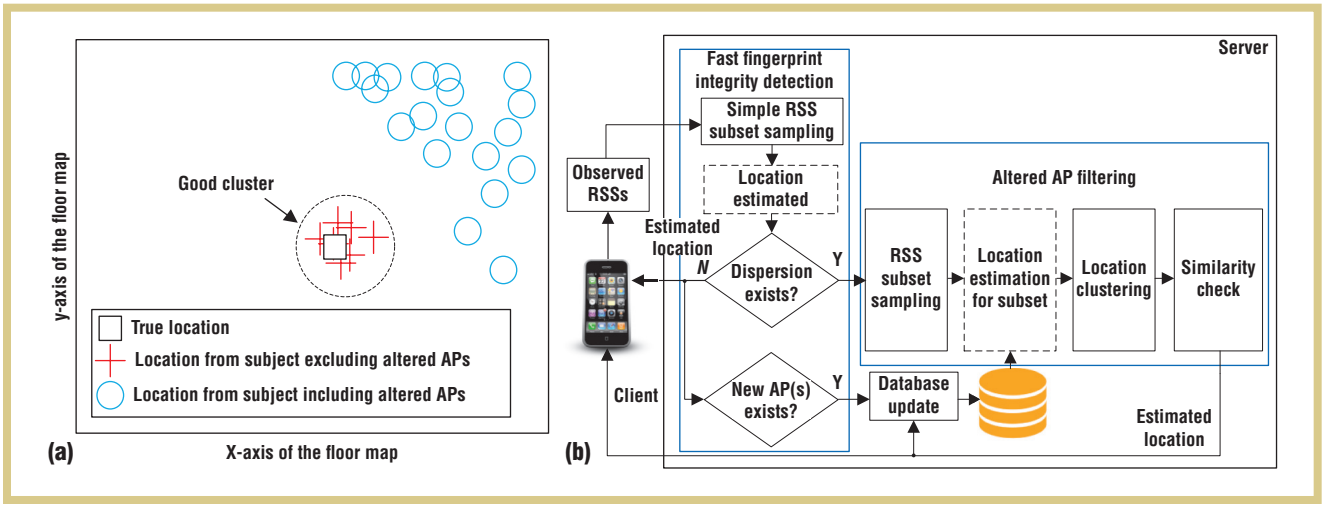


Figure 2. The core of Chameleon system: (a) an observation of subset location estimation, and (b) an overview of the Chameleon system.

fingerprint-integrity detection is too fast and early to detect whether the fingerprint measured by the client at a location is integral. If so, localization can be conducted efficiently. Otherwise, the more costly AP filtering would be executed.

Altered-AP Filtering

In this phase, Chameleon randomly samples subsets of APs obtained from the RSS reported by the client, creating the *RSS subset sampling*. It then estimates a location for each of these subsets. For those subsets excluding altered APs, their estimated locations would cluster together. The remaining subsets generate locations spreading around the region. The clustering algorithm (see Figure 3) and a fingerprint similarity check are then conducted on these locations to identify the cluster (or clusters) consisting of the unaltered APs. This yields the accurate location of the client.

Fingerprint Database Update

Traditionally, the measured RSS vector of the client is discarded after the localization decision. In Chameleon, the RSS is used in the fingerprint update phase, where the RSS containing the altered AP (or APs), together with the estimated location, is used to update the radio map as it captures new environmental signal characteristics.

Fast Detection for Fingerprint Integrity

AP signal alteration isn't usually beyond noise level. Even when this happens, the affected area is only the area covered by the altered AP. Therefore, it's more likely that the RSS received doesn't contain altered APs. We thus need a fast algorithm to detect fingerprint integrity early on so that the user location is efficiently estimated and returned based on the collected RSS. However, the algorithm shouldn't lead to frequent fingerprint updates when there are transient signal changes in a dynamic environment.

Under the relatively uncommon case that fingerprint inconsistency is detected (altered APs exist in the measured RSS vector), then the computationally more expensive altered-AP filtering should be executed to find the user's location. Here, we outline the process for fast fingerprint-integrity detection.

Simple RSS Subset Sampling

We first obtain a few sizable subsets (say, five or six) of the measured RSS, and we use these subsets to estimate the client's location. Although any subsets will do, for simplicity, we use the following to cover all of the APs:

- the full original RSS vector;
- two subsets of similar size, obtained by dividing the measured RSS vector into two parts in the middle; and
- three subsets of similar size, obtained by dividing the measured RSS vector into three equal parts.

Therefore, only six subsets are involved in the location estimation step.

Location Estimation

Using the subsets generated, Chameleon estimates the user locations (based on any localization algorithm). In other words, a set of locations are generated.

Dispersion Detection

For the locations generated, Chameleon computes their pair-wise Euclidean distances. If the average distance is lower than a certain threshold, denoted as t_d, the estimated locations of the RSS subsets are very close to each other, and Chameleon concludes there are no altered APs. In this case, the centroid of the estimated locations is returned as the client position.

On the other hand, if the average distance is above t_d, the dispersion indicates that some AP signals might have

Input: Received RSS vector **v** from client.
Locations of RPs.
Signal vectors of all RPs.
Number of RSS subsets M to be generated.
Output: Location estimation $\langle \hat{x}, \hat{y} \rangle$ for client.
/* RSS subset sampling. */
1 Set of RSS subsets $\Psi \leftarrow \{\}$.
2 **while** $|\Psi| \leq M$ **do**
3 RSS subset $\mathbf{v}_s \leftarrow \{\}$.
4 **for** *each* received *AP* A_l **do**
5 Toss a coin for A_l.
6 **if** *the coin is head* **then**
7 Add $\{A_l : T_l\}$ into $\mathbf{V}_s$.
8 **end**
9 **end**
10 **if** $|\mathbf{v}_s| \geq \gamma$ **then**
11 Extend $\mathbf{v}_s$ into $\tilde{\mathbf{V}}_s$
12 Add $\tilde{\mathbf{V}}_s$ into Ψ.
13 **end**
14 **end**
/* Location estimation. */
15 Set of location estimations $\Omega \leftarrow \{\}$.
16 **for** *each RSS subset* $\tilde{\mathbf{v}}_s$ in Ψ **do**
17 Compute estimated location $<x_s, y_s>$ for $\tilde{\mathbf{V}}_s$.
18 Add $<x_s, y_s>$ into Ω.
19 **end**
/* Clustering the location estimations. */
20 Apply clustering algorithm on locations Ω.
/* Cluster similarity check. */
21 Set of cluster similarities $\Theta \leftarrow \{\}$.
22 **for** each cluster c in formed clusters **do**
23 Calculate centroid location $<x_c, y_c>$ of cluster c.
24 Calculate location distance between $<x_c, y_c>$ and the RPs.
25 Pick Q nearest RPs from $<x_c, y_c>$.
26 Calculate cluster similarity Θ^c between RSS subsets in cluster c and these Q nearest RPs.
27. Add Θ^c into Θ.
28. **end**
29. Pick C^* with maximum cluster similarity in Θ.
30. $\langle \hat{x}, \hat{y} \rangle \leftarrow$ Centroid of C^*

Figure 3. The Chameleon system's altered access point (AP) filtering algorithm. For a received RSS sample V, multiple subsets V_s are first generated by random sampling (lines 1–14). The K-Nearest-Neighbor (K-NN) algorithm is implemented to estimate the locations for these subsets (lines 15–19). Clustering is then conducted on these locations based on *k*-means algorithm (line 20). By checking cluster similarity, Chameleon identifies the good cluster, where the target is likely located (lines 21–29).

changed. Chameleon then moves to the more expensive phase, altered-AP filtering, to compute the location of the users. In our deployment, we set the threshold as $t_d = \eta \cdot \bar{e}$, where η is a predefined parameter and $\bar{e}$ is the mean localization error of offline training target samples (without altered APs).

Discovery of New APs

For a client location, if his or her RSS consists of an AP that has not been in the fingerprint database, a new AP is found. In this case, we need to update the database by adding the newly discovered AP and its signal into the database.

Indoor Localization by Filtering Altered Aps

Given the decision from fast detection that there might be altered APs, Chameleon further conducts subset localization and clustering to locate the target and filter out altered APs.

RSS Subset Sampling

Let N be the total number of APs detected in the whole site, and set $\mathrm{A} = \{1, \ldots, N\}$ be the index of the APs. We denote the measured RSS sample of the target (client) as

$$\mathrm{V} = \{\phi_1, \phi_2, \ldots, \phi_{l, \ldots}, \phi_L\},$$

where L is the number of APs received by the target ($1 \leq L \leq N$), and $\phi_l = \{A_l : \mathrm{T}_{\mathrm{A}_l}\}(1 \leq l \leq L)$ is the couplet indicating the received AP index $A_l \in \mathbf{A}$ and its corresponding signal strength T_{A_l}(mW). Given **V**, we further define its *extended signal vector* $\tilde{\mathrm{V}}$ by padding those undetected APs with very low value (say, 0)—that is,

$$\tilde{\mathrm{V}} = [T_1, T_2, \ldots, T_i, \ldots, T_N],$$

where $T_i = 0$ if $i \neq A_l$ for $\forall l \in \{1, \ldots, L\}$.

To filter altered APs in the measured RSS sample, we first generate RSS subset $\mathbf{V}_s$ by randomly selecting APs from **V**—that is, $\mathbf{V}_s \subset \mathbf{V}$. For each generated RSS sample $\mathbf{V}_s$, we then form its extended vector $\tilde{\mathbf{V}}_s$ analogously as **V** to $\tilde{\mathbf{V}}$.

For efficiency, we can't generate all the possible subsets of **V**. Therefore, we generate a certain number of subsets that are random samples of the exhaustive set (that is, uniformly chosen from the power set). This linear sampling works as follows. For each detected AP, we toss a fair coin to decide whether to include it into the subset. As the subsets will be used for localization, each subset shouldn't consist of too few APs (we used

a minimum of three APs in our study). Otherwise, the subset will be discarded.

Note that because the subset size and the included APs are random, some subsets will likely exclude altered APs. These subsets would result in a cluster, providing accurate location estimation for the client.

Localization Algorithm

We perform location estimation for all generated RSS subsets. As mentioned before, Chameleon is independent of—and thus can be integrated with—any fingerprint-based localization algorithms. For concreteness, we use the *K*-Nearest-Neighbor (*K*-NN) to estimate the user location for each RSS subset.

In *K*-NN, we find the *K* nearest RPs whose signal strengths closely match $\tilde{\mathrm{V}}_s$. That is, it finds the *K* RPs that have the smallest Euclidean distances from $\tilde{\mathrm{V}}_s$. Mathematically, we denote the RSS sample at an RP as **U**, and we denote its corresponding extended vector $\tilde{\boldsymbol{U}}$, given by

$$\tilde{\boldsymbol{U}}=[R_1,\ R_2,...,\ R_N].$$

With $\tilde{\boldsymbol{U}}$ and $\tilde{\mathrm{V}}$, their Euclidean distance *D* is defined as

$$D^2=\sum_{l=1}^{N}\left(T_l-R_l\right)^2.$$

After finding the *K* RPs, we normalize their signal strength distances as weights of the corresponding RPs. The location is then estimated as the weighted average of these RP locations. More precisely, let D_q be the distance between RP *q* and the client signal ($1 \le q \le K$). The estimated location $\langle \hat{x}_s, \hat{y}_s \rangle$ for a subset $\mathbf{V}_s$ is expressed as

$$\langle \hat{x}_s, \hat{y}_s \rangle = \sum_{q=1}^{K} \omega_q \langle x_q, y_q \rangle,$$

where $<x_q, y_q>$ is the RP *q*'s position, and ω_q is the normalized weight of RP *q* given by

$$\omega_q = \frac{\frac{1}{D_q}}{\sum_{q=1}^{K}\left(\frac{1}{D_q}\right)}.$$

Location Clustering

The locations estimated from RSS subsets without any altered AP tend to cluster together and are called the *good cluster*. The locations of subsets consisting of some altered APs tend to disperse on the floor map. To identify the cluster excluding altered APs, we use a clustering algorithm on the calculated locations of the subsets.

We employ the *k*-means clustering algorithm to find the cluster that excludes altered APs. (Chameleon is independent of, and thus compatible with, other clustering algorithms, such as the algorithm proposed by Brendan Frey and Delbert Dueck[6]). Given a set of 2D data points, we partition them into *k* clusters based on their Euclidean distance.

Similarity Check to Find the Client Location

After the clustering algorithm, we use the similarity check to distinguish the good cluster from the other clusters, thereby identifying the client's location. Because most RSS subsets in the good cluster exclude altered APs, they show stronger similarity with the fingerprint in the database. Therefore, by calculating the similarity between the cluster and the fingerprint database (called *cluster similarity*), we can determine which cluster is the good one for client localization.

During this stage, we implement cosine similarity to measure the closeness of two vectors. Given any two extended signal vectors $\tilde{\boldsymbol{U}}$ and $\tilde{\mathrm{V}}_s$, we compare their cosine similarity, defined as

$$\theta\left(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{V}}_s\right)=\frac{\tilde{\boldsymbol{U}}\cdot\tilde{\boldsymbol{V}}_s}{\left|\tilde{\boldsymbol{U}}\right|\left|\tilde{\boldsymbol{V}}_s\right|}. \quad (1)$$

For each cluster, we select several nearest RPs according to the cluster's centroid location. Then, for each RSS subset in the cluster, we sum up its cosine similarity with all these RPs. Finally, we compute cluster similarity between all subsets in the cluster and RPs—that is, the average of all the similarity between subsets and the nearest RPs. For cluster *c*, denote the fingerprints of the nearest *Q* RPs as $\left\{\tilde{\boldsymbol{U}}_q^c \mid 1 \le q \le Q\right\}$, and its *s*th RSS subset as $\tilde{\mathbf{V}}_s^c\left(1 \le s \le S^c\right)$. Then, based on Equation 1, the cluster similarity θ^c is calculated by

$$\theta^c=\frac{1}{S^c}\sum_{s=1}^{S^c}\sum_{q=1}^{Q}\theta\left(\tilde{\boldsymbol{U}}_q^c,\tilde{\boldsymbol{V}}_s^c\right).$$

The cluster with the highest similarity θ^c will be selected as the good one, and its centroid will be returned as the potential client location.

Database Update with the Client RSS

After estimating the client's location, we use the RSS collected by the client to update the database. This is possible because the collected signal strengths—in particular, the altered ones—have captured environmental changes. The measured RSS, along with the location, can be used to update that location's signature (fingerprint).

We update the fingerprint database as follows. We denote the distance between the estimated client location and his or her nearest RP as *d*. Let t_u be a threshold distance (say, a few meters) for the update, above which the signal doesn't observably change.

If $d < t_u$, we conclude that the nearest signature has been changed due to altered APs. In such a case, we use the measured RSS vector α from the client to update the RP's signal strength. If a new AP is detected, a new entry is initiated at the RP and its corresponding RSS is added to the vector. For the other identified APs, let $0 \le \alpha \le 1$ be the weight indicating how likely the new data affects the stored signature ($\alpha = 0$ means that a new incoming RSS doesn't affect the signature, and vice versa).

Then we denote the current signature as $\hat{\mathbf{V}}$ and update it based on the new client data $\tilde{\mathrm{V}}$ such that

$$\hat{\mathrm{V}} \leftarrow (1-\alpha)\ \hat{\mathrm{V}}+\alpha\ \tilde{\mathrm{V}}.$$

For the case that $d > t_u$, $\tilde{\mathrm{V}}$ is discarded and the radio map isn't updated.

Experimental Results

We implemented Chameleon and evaluated its performance through experiments on our campus, The Hong Kong University of Science and Technology (HKUST), and at the Hong Kong International Airport (HKIA).

Experimental Settings and Comparison Metrics

We developed an Android APP to provide indoor localization service at both HKUST and HKIA. During data preprocessing, we filtered out the mobile APs tethered by smartphones and combined virtual APs (VAPs).[7] For the localization decision, we filtered out the APs whose RSS in the whole survey site was lower than minus 90 dBm.

We conducted experiments on two sites, the first of which was the HKUST campus atrium, an indoor atrium (2,700 m^2) in which data from 239 RPs was collected (on 18 May 2014). The RPs were taken at three-meter intervals. Client RSSs were taken in the middle of RPs. Overall, 140 physical APs were selected for testing. At each RP, a client could measure 28 distinctive physical APs on average (5 APs of signals above -60 dBm). Each target could detect 16 physical APs on average.

The second experiment site was the HKIA departure gate floor, a 27,300 m^2 area in which data from 1,364 RPs was collected on 2 January 2014. The RPs were taken at five-meter intervals. We selected 350 physical APs in the survey site for experiment. Many of the installation points of the APs were outside our survey area (though we detected their signals in the area). Each RP measured 47 distinctive APs on average (4 APs of signals above –60 dBm), and each target could detect 28 APs. (Details of the floor plans appear elsewhere.[8])

During the site survey and target collection, the detected APs were under uncoordinated deployment from various parties at different floors or locations. The experimental sites in the airport and campus atrium included a large open indoor space, where the signal attenuation was relatively low compared with office buildings. Thus, we could measure many APs in both sites.

Unless otherwise stated, we used the following parameters in our experiments:

- We set K to 10 in K-NN location estimation (based on our empirical studies, which found that K = 10 resulted in few errors and required little computation).
- We selected nine nearest RPs for each cluster in cluster similarity calculation.
- We used four clusters for k-means clustering.
- We set the threshold of fast fingerprint-integrity detection to $\eta = 1.5$ for both HKUST and HKIA ($\overline{e} = 3.9$ m for 1,200 offline training samples in HKUST and $\overline{e} = 9.95$ m for 1,600 samples in HKIA).
- We generated 200 subsets for each subset sampling.
- We set a to 0.5 in the database update.
- The distance threshold was t_u = 8 m for the database update.

In HKUST, after we collected the fingerprint, we reduced the power of two APs in the whole survey site to study Chameleon performance. For HKIA, four APs were altered for evaluation. The altered APs were introduced by setting the power P to βP (mW), where b was 0.4 (by default); we conducted 30 trials overall at each client position.

We evaluated the localization performance and signal adaptivity using the following metrics.

The first metric was the *accuracy of fast fingerprint-integrity detection*. Given overall E testing cases (targets), we let *TP (true positive)* be the number of correct classifications with actually changed APs, and *TN (true negative)* be that with unchanged APs. Similarly, we let *FP (false positive)* be the number of misclassifications whose measured APs were actually unchanged and *FN (false negative)* be that with actually changed APs. Then, $E = TP + TN + FP + FN$. We defined the true detection rate (TDR) as

$$\mathrm{TDR} = \frac{TP + TN}{E},$$

while the false alarm rate (FAR) was defined as

$$\mathrm{FAR} = \frac{FP}{FP + TN}.$$

The true positive rate (TPR) was given by

$$\mathrm{TPR} = \frac{TP}{TP + FN},$$

while true negative rate (TNR) was

$$\mathrm{TNR} = \frac{TN}{TN + FP}.$$

The next metric was the *localization error e*, given by the Euclidean distance

$$e^2 = (x - \hat{x})^2 + (y - \hat{y})^2,$$

where (x, y) is the true location and $(\hat{x}, \hat{y})$ is the estimated location.

The third metric was the *localization weight of AP l*—that is, the percentage of subsets with AP l over all the subsets. M_l was the number of subsets in the cluster including AP l, and $|V_s|$ was the total number of APs included in the sth RSS subset. We denoted the number of subsets to be generated as M. Then the weight of AP l was defined as

$$W_l = \frac{M_l}{\sum_{s=1}^{M} |V_s|}.$$

We compared Chameleon with the Bayesian method and Cosine similarity algorithms. Bayesian algorithm leverages the maximum likelihood to estimate the location.[2] It first calculates the probability distribution of the RSS value at each RP. Given a target RSS vector, it computes the overall probability of the vector at each RP and finds the one with the maximum likelihood to determine the target location. The Cosine similarity algorithm uses signal similarity to estimate loca-

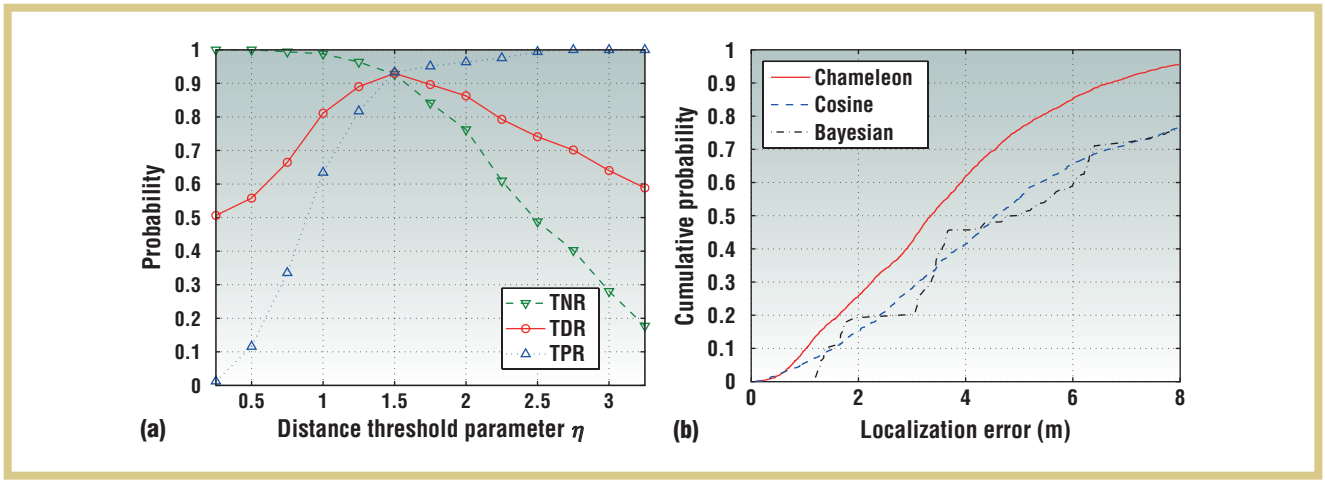


Figure 4. Overall performance of Chameleon at The Hong Kong University of Science and Technology (HKUST): (a) the true positive rate (TPR), true detection rate (TDR), and true negative rate (TNR) versus the distance threshold parameter; and (b) the cumulative probability distribution of localization error in fast fingerprint-integrity detection.

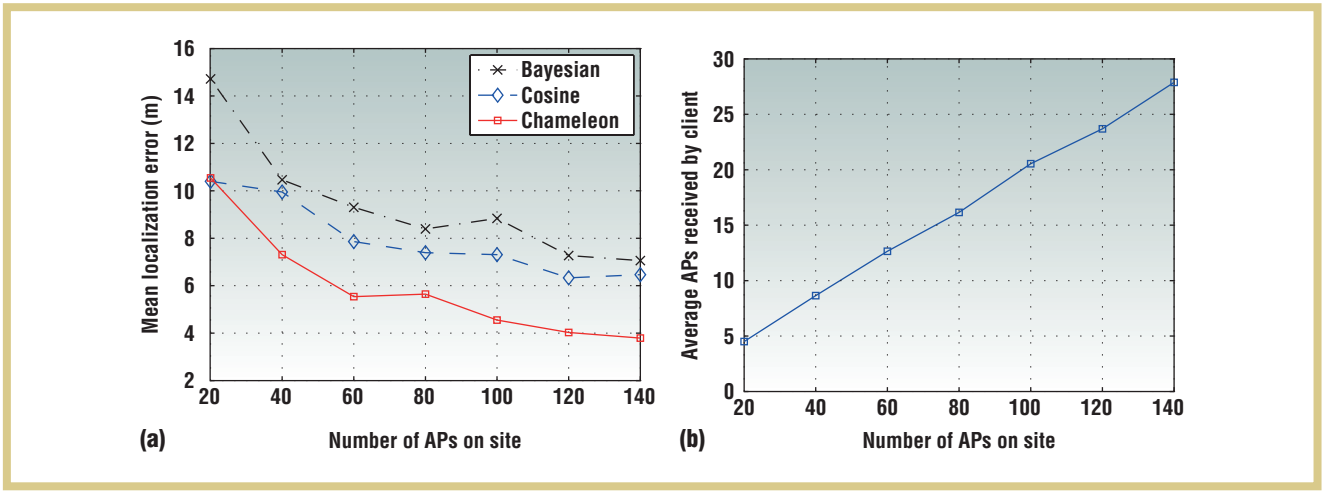


Figure 5. Experimenting with the AP number (at HKUST): (a) the localization error versus the number of APs at the site, and (b) the number of APs received at the client's location versus the number of APs at the site.

tion. The similarity between a received signal vector and the RP fingerprint is calculated according to Equation 1. Then the location is computed based on the K-NN algorithm.[1,9]

Illustrative Results

Figure 4 shows the overall performance of Chameleon at HKUST. Figure 4a shows TPR, TDR, and TNR versus the distance threshold parameter η in fast fingerprint-integrity detection. We conducted the testing over 1,200 targets collected at the HKUST campus, half of which were positive cases (we randomly altered two of the detected APs at each target). By varying η on some offline samples, we found the threshold t_d with the optimal dispersion detection.

In general, TDR first increased and then decreased, while TPR decreased and TNR increased. When η was small, t_d was small. The fast detection was sensitive, and many targets were classified as "AP alteration," leading to high FP. TPR was thus high while TNR was low. As t_d increased, the influence of transient signal fluctuation decreased and FP was thus reduced. As t_d further increased, TDR decreased, because a loose threshold could also classify AP alteration as "unaltered," leading to more FN and lower TPR. In HKUST, the optimal TDR was 93.3 percent, and FAR was 6.8 percent.

We also collected 1,600 targets at HKIA and conducted a similar experiment. The optimal TDR and FAR at HKIA was 93.5 percent and 7.7 percent, respectively. For HKUST and HKIA, we set η = 1.5 as the default during deployment.

Figure 4b shows the cumulative probability of localization error. Chameleon achieved much better localization accuracy than the other two algorithms, because it pruned the altered APs before

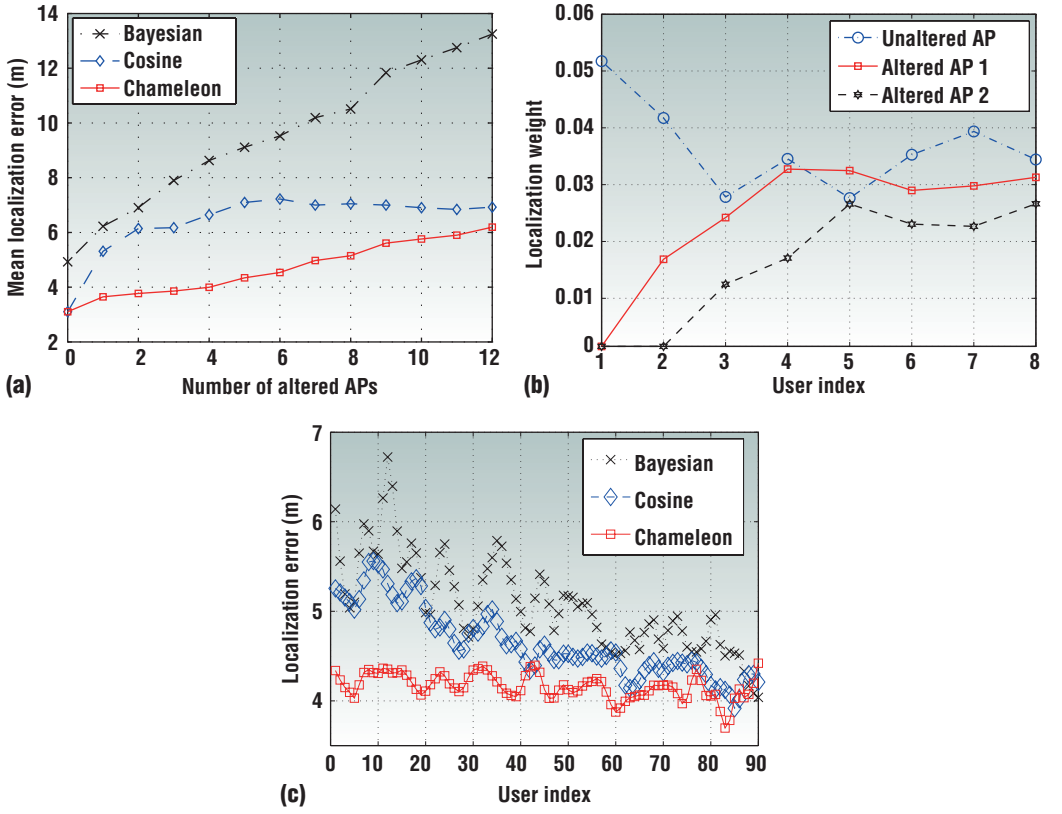


Figure 6. Experiment on adaptivity to signal change (HKUST): (a) the localization error versus the number of altered APs, (b) the localization weight of APs versus the user index, and (c) the localization error with updated fingerprints.

localization. The other approaches suffered from higher error because altered APs led to a dispersed set of matching RPs, and the client was less likely to be mapped to the correct region.

Figure 5 shows the impact of the AP number on localization accuracy. We randomly turned on some of the APs to conduct experiments. Figure 5a shows the average localization error versus the number of APs on the site. Clearly, localization error decreased with the number of APs, because more APs mean better differentiation of fingerprints and thus better localization.

Chameleon achieved much lower localization error because of its altered-AP filtering. By clustering the estimations and finding the good cluster, it reduced the chance of mapping the client to a wrong location. There was diminishing return as we increased the number of APs in the site (approximately 80 to 100 in this experiment), because signal measurement noise became the limiting factor as the AP number increased.

Figure 5b shows the average AP number received by a client versus the number of APs on the site. The greater the number of APs, the more APs there are for the client to detect. We see that a low localization error can be achieved when there is a sufficient number of APs (approximately 20) near the client. Too few detectable APs would lead to poor performance.

We also conducted experiments to study the learning process or database update of Chameleon. Figure 6a shows the localization error versus the number of altered APs. Localization errors of both Cosine and Bayesian increased quickly with the increase of altered APs. Chameleon showed better robustness against altered APs, mainly because it filtered out the altered APs through clustering.

Figure 6b shows the weight (importance) of APs in localization versus the user update index. We show the weights of the two altered APs—AP 1 and AP 2—and an unaltered AP 3 over time. Initially, only the unaltered APs were used for localization. The localization weights of altered APs were low at the beginning, because the APs were filtered out in the good cluster. The weights of AP 1 and AP 2 gradually increased with more incoming user records, showing that the database gradually adapted to the altered signals. Chameleon was effective, because it identified the altered APs and updated the database accordingly.

Figure 6c shows the localization errors against the index of fingerprint

updates. We observed remarkable improvement with the Cosine and Bayesian algorithms when the fingerprint database was updated by the Chameleon scheme. For Chameleon, the fingerprint update reduced the discrepancy between the database and the environment. It reduced the dispersion of fast detection in subsequent user localization, thereby reducing the need for the more expensive second phase of subset sampling.

We also conducted various other experiments at HKIA. As the results were quite qualitatively similar, we don't include them here due to space constraints. Further details appear elsewhere.[8]

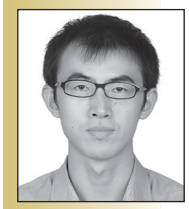
In our future research studies, we will further integrate Chameleon with large-scale crowdsourcing systems for more ubiquitous localization scenarios. In this way, the fingerprint-based localization system will evolve over time and adapt to the environmental changes. P

REFERENCES

1. D. Han et al., "Building a Practical Wi-Fi-Based Indoor Navigation System," *IEEE Pervasive Computing*, vol. 13, no. 2, 2014, pp. 72–79.
2. P. Mirowski et al., "Probabilistic Radio-frequency Fingerprinting and Localization on the Run," *Bell Labs Technical J.*, vol. 18, no. 4, 2014, pp. 111–133.
3. C. Feng et al., "Received-Signal-Strength-Based Indoor Positioning Using Compressive Sensing," *IEEE Trans. Mobile Computing*, vol. 11, no. 12, 2012, pp. 1983–1993.
4. S. He and S.-H. Gary Chan, "Wi-Fi Fingerprint-Based Indoor Positioning: Recent Advances and Comparisons," *IEEE Comm. Surveys Tutorials*, vol. 18, no. 1, 2016, pp. 466–490.
5. Martin A. Fischler and Robert C. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *ACM Comm.*, vol. 24, no. 6, 1981, pp. 381–395.
6. B.J. Frey and D. Dueck, "Clustering by Passing Messages between Data Points," *Science*, vol. 315, no. 5814, 2007, pp. 972–976.
7. E. Martin et al., "Precise Indoor Localization Using Smart Phones," *Proc. 18th ACM Int'l Conf. Multimedia*, 2010, pp. 787–790.
8. B. Ji, "Chameleon: Survey-Free Updating of Fingerprint Database for Indoor Localization," master's thesis, The Hong Kong University of Science and Technology, Hong Kong, July 2014.
9. S. Han et al., "Cosine Similarity Based Fingerprinting Algorithm in WLAN Indoor Positioning Against Device Diversity," *Proc. 2015 IEEE Int'l Conf. Comm.* (ICC), 2015, pp. 2710–2714.

the AUTHORS

Suining He is currently working toward his PhD in the Department of Computer Science and Engineering at The Hong Kong University of Science and Technology (HKUST). His research interests include indoor localization and mobile computing. He received a BEng in mechanical engineering and automation from Huazhong University of Science and Technology (HUST). He is a student member of IEEE, the IEEE Communications Society, and the IEEE Computer Society. Contact him at sheaa@cse.ust.hk.



Bo Ji graduated from The Hong Kong University of Science and Technology (HKUST). His research interests include mobile computing and indoor. Ji received an MPhil in computer science from HKUST. Contact him at bji@cse.ust.hk.

S.-H. Gary Chan is a professor and undergraduate programs coordinator at the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST). He is also the director of Sino Software Research Institute at HKUST. His research interests include multimedia networking, wireless networks, mobile computing, and IT entrepreneurship. Chan received a PhD in electrical engineering from Stanford University. He is a member of the honor societies Tau Beta Pi, Sigma Xi, and Phi Beta Kappa. Contact him at gchan@cse.ust.hk.