Slide: Towards Fast and Accurate Mobile Fingerprinting for Wi-Fi Indoor Positioning Systems

Kongyang Chen, Chen Wang, *Member, IEEE*, Zhimeng Yin, Hongbo Jiang, *Senior Member, IEEE*, and Guang Tan, *Member, IEEE*

***Abstract*— The deployment of Wi-Fi fingerprint-based indoor positioning systems is severely hindered by the lack of an efficient and low-cost way to establish a signal fingerprint database. In this paper, we present a novel fingerprinting method, *slide*, that can collect fingerprints in a fast and accurate way. Slide uses a commodity flashlight and a smartphone to achieve *linear positioning*. This allows automatic mapping from the received signal strength to the position on a line, serving as a building block for fingerprinting in general environments. Slide also features a channel-based scanning method, which acquires fingerprint location after each Wi-Fi channel scanning, to mitigate the *fingerprint misalignment* problem found in the general mobile fingerprinting. Quantitative analysis and experimental results show that slide is faster than the manual fingerprinting method by up to an order of magnitude with comparable positioning accuracy, and is also more efficient than state-of-the-art mobile fingerprinting methods.**

***Index Terms*— Indoor positioning, WiFi scan, fingerprint misalignment, light positioning.**

I. Introduction

WIFI based positioning is considered a promising solution for indoor location-based services [1]–[3]. The standard way of positioning is to compare the received signal strength (RSS) against a pre-stored RSS fingerprint database, which provides a mapping from the RSS to the position. The process of establishing such a mapping, also known as training, hinges on a method to find out the current position of a device when it is collecting RSS values. In this paper we call such a task *training stage positioning* (TSP). The simplest and most used fingerprinting method is by manual operations [4]–[7]. In this method, a user holds a device at *reference spots* (often grid points in an area) to measure the RSSs of access points (APs) in range. Due to fluctuation of WiFi signals, a certain amount of time (e.g., 30 seconds) is needed at every spot to collect enough RSS samples to generate a relatively stable average. The TSP task is then accomplished manually, for example, with the help of pre-setup position marks on the ground.

One major concern of the manual fingerprinting method, however, is that it is quite slow and costly, as it relies on manual operation. While this may be acceptable in small scenarios such as an office environment, as assumed in many previous studies such as RADAR [4], Horus [8], WILL [9], and Zee [10], it is prohibitively expensive in large environments such as airport terminals and shopping malls, which are perhaps the places where indoor location based services (LBSs) are most needed.

Recently, there has been a considerable amount of attention paid to mobile fingerprinting techniques [10]–[14], i.e., collecting fingerprints on the move. These approaches often require the collaboration of people who collect RSS fingerprints, carrying mobile devices around, while the position information is not from the explicit user input, but inferred from motion sensors. With the opportunistic strategy, the fingerprints are collected implicitly during users' daily activities. Though these methods largely obviate the need for dedicated efforts, they rely on an incentive mechanism to recruit participants to make contributions. In practice, the motivation for free or low-paid contributions may not be strong enough to attract stable cooperation from large numbers of ordinary users. This means that the fingerprinting task will end up falling on a relatively small group of users, e.g., the staff members of those places. Given the opportunistic and iterative nature of the process, it is likely to result in slow convergence of the database and hence high cost in terms of setup time.

In this paper we propose a new mobile fingerprinting method, dubbed as *Slide*, that enables *fast setup* of

Manuscript received June 27, 2017; revised September 19, 2017; accepted November 24, 2017. Date of publication November 28, 2017; date of current version January 8, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61379135, Grant 61572219, Grant 61502192, Grant 61732017, Grant 61271216, Grant 61772509, Grant 41701479, and Grant 61471408, in part by the China National Key Research and Development Program under Grant 2016YFB0502202, in part by the Shenzhen Overseas High-Level Talents Innovation and Entrepreneurship Funds under Grant KQCX-20140520154115026, in part by the Shenzhen Fundamental Research Program under Grant JCYJ20140610151856733, and in part by the Shenzhen Scientific Research and Development Funds under Grant CXZZ20151117161747567. The associate editor coordinating the review of this paper and approving it for publication was Dr. Thilo Sauter. *(Corresponding author: Chen Wang.)*

K. Chen is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China, and also with the Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: ky.chen@siat.ac.cn).

C. Wang and H. Jiang are with the School of Electronic Information and Communications, Huazhong University of Science and Technology, Wuhan 430074, China (e-mail: cwangwhu@gmail.com; hongbojiang2004@gmail.com).

Z. Yin is with the Department of Computer Science, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: yinzhimeng@gmail.com).

G. Tan is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China (e-mail: guang.tan@siat.ac.cn).

Digital Object Identifier 10.1109/JSEN.2017.2778082

TABLE I

CHARACTERISTICS OF DIFFERENT FINGERPRINTING METHODS. v REFERS TO VEHICLE' MOVING SPEED

Method	Manual/Mobile	Labor cost	Equipment cost	General topology	Fingerprint mismatch error
Manual	Manual	High	Low	Yes	0
WILL [9]	Mobile	Low	Low	No	$\approx v$
Zee [10]	Mobile	Low	Low	No	$\approx v$
LiFS [11]	Mobile	Low	Low	No	$\approx v$
UnLoc [12]	Mobile	Low	Low	No	$\approx v$
SLAM [17]–[19]	Mobile	Low	High	Yes	$\approx v$
Walkie-Markie [13]	Mobile	Low	Low	(unknown)	$\approx v$
Slide	Mobile	Low	Low	Yes	$\approx 0.1v$

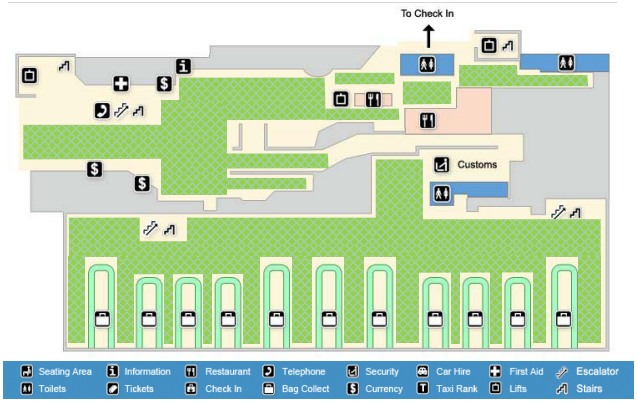


Fig. 1. An airport terminal scenario [15] where Slide can be used to efficiently establish a fingerprint database for the main (green) areas.

fingerprint database, by using vehicles rather than humans as mobile data collectors. The idea, however, is not merely to use a machine to replace humans as a device carrier, but to solve the TSP problem that lies at the heart of the fingerprinting task. Slide does not insist on fully automatic TSP, which requires a separate infrastructure for positioning. Instead, it focuses on the so-called *linear positioning*, a subproblem that is more tractable (yet non-trivial) to users with only simple and inexpensive equipments. Slide offers a novel solution using the built-in light sensor on a regular smartphone and a commodity LED flashlight, in light of the drawbacks of alternative smartphone sensors for the same purpose (see the discussion in Section II).

Slide is intended to be a complementary solution to existing fingerprinting techniques. For example, it can serve as a fast solution for the initial establishment of a fingerprint database for the main area of a large place, while manual or crowdsensing based methods could be used to fill in the gaps in the data, or to update it in response to environmental changes. In this sense, Slide is particularly suitable for public places with large free spaces such as atria and hallways. For instance, in an airport terminal as in Fig. 1, Slide can help establish a fingerprint database quickly for the main areas, while manual or crowdsensing based methods can be used to fingerprint remaining corners or small areas.

Overall, Slide can provide the following features:

1) Speed. Slide greatly exceeds existing approaches in fingerprinting efficiency. Consider a 40 meters long straight path, on which 40 reference spots are selected. With manual fingerprinting, each spot will take about 30 seconds,[1] and thus it needs 1200 seconds to complete fingerprinting. In contrast, in general, Slide takes 6 trips for RSS samplings to converge satisfactorily. Assuming a vehicle with an average speed 4m/s, Slide needs only 10 seconds to complete one trip, and 120 seconds to generate a stable database, reducing the time by a factor of 10. Slide is also faster than human walking (normally less than 1.5 m/s) and some robots (e.g., the Roomba at 0.3 m/s [16]).
2) Accuracy. Slide performs comparably with the manual fingerprinting method and outperforms existing mobile schemes in terms of positioning accuracy. We reveal an inherent *fingerprint misalignment* problem in general mobile fingerprinting methods. The problem arises from the one-second scanning delay of a standard 802.11 driver, during which the vehicle may have moved a few meters, causing mismatch between the RSSs and the corresponding position. Slide leverages a channel-based scanning protocol, which acquires fingerprint location after each WiFi channel scanning instead of traditional all channel scanning, to provide more prompt response of RSSs. Our method can reduce the misalignment error by an order of magnitude and introduces minimal loss to positioning accuracy.
3) Practicality. Slide uses off-the-shelf components, namely flashlights and light sensors, along with any vehicle that can carry a smartphone, to achieve linear positioning. Without relying on heavy on-site measurements and calibrations, and without the requirement of specialized and expensive equipments (e.g., a SLAM robot [17]–[19]) or infrastructure, Slide offers an easily implementable solution for WiFi fingerprinting.

The remainder of this paper is organized as follows. Section II provides a brief review of previous works. Section III gives an overview of the light-based linear positioning system. Section IV discusses the fingerprint misalignment method. Section V presents two typical cases of our localization system. Section VI shows the our evaluation results. Finally, Section VII concludes this paper.

II. RELATED WORK

In this section we discuss three main techniques for WiFi fingerprinting. Table I compares several related approaches.

[1]Note that for improved performance, a user may take multiple fingerprints at a spot, each corresponding to a different orientation of the device. For simplicity, we assume only a single fingerprint at each spot. This does not affect the validity of our performance comparison.

We also sketch indoor localization works with other sensors (i.e., visible light).

A. Human Walking and Manual Labeling

In RADAR [4], Horus [8], PlaceLab [5], and SurroundSense [6], a user holds a smartphone at each fingerprint spot and waits a certain amount of time (e.g., 30 seconds) for the WiFi module to collect enough RSS samples, which are used to form a fingerprint pattern. The user's current position is labeled manually by some means of distance measurement, for example using a ruler or pre-setup marks on the ground. The interval between the fingerprint spots is typically one or a few meters in order to achieve good accuracy. In a medium sized or large environment, the fingerprinting process is laborious and time consuming. OIL [20] exploits the collaboration of a group of people to collect and label fingerprints. This method, however, still suffers from a long set-up process, and is susceptible to mislabeling which is likely to happen with untrained participants. Our paper provides a fast and accurate fingerprint sampling method with a smartphone mounted on a mobile vehicle, which simultaneously records the WiFi fingerprint and its correct location label.

B. Human Walking and Automatic Labeling

Following the spirit of crowdsensing, a number of methods attempt to have a group of people collect RSS fingerprints collaboratively in their daily activities. The TSP problem is solved by combining accelerometers, gyroscopes and magnetic sensors [21]–[23] in mobile devices. In WILL [9], the collected fingerprints are mapped to floor positions considering WiFi signal strength and connectivity of rooms. A similar idea is used in LiFS [11], where the relationship between WiFi signal space and stress-free floor plan is exploited to automatically map the fingerprints to positions. Zee [10] utilizes a particle filter to determine the walking trace by combining data from motion sensors. UnLoc [12] and SemanticSLAM [24] propose to identify organic landmarks that can help to solve the TSP problem. The recent work, Walkie-Markie [13], captures the changing pattern of WiFi signal strength during human movement. Combined with users' motion patterns, Walkie-Markie is able to generate the indoor pathway which can be used to label WiFi fingerprints automatically.

Though these methods manage to reduce human efforts in the fingerprinting process, they implicitly rely on the existence of distinguishing structural or signal features, such as corridors, corners, and doors, so as to remove ambiguities in trace mapping and position determination. This assumption, however, may not hold in some circumstances. Fig. 2(a) shows a scenario where two traces follow similar movement patterns with the only difference in their turning points. Due to the error of step size based distance estimation, the turning points can easily be confused, and thus the subsequent traces are more likely to be wrongly determined. Fig. 2(b) shows another example of trace mismatching due to heading error which is very common indoors. These ambiguities could be reduced by identifying unique environmental features such as special radio patterns, which in turn introduces extra assumptions and

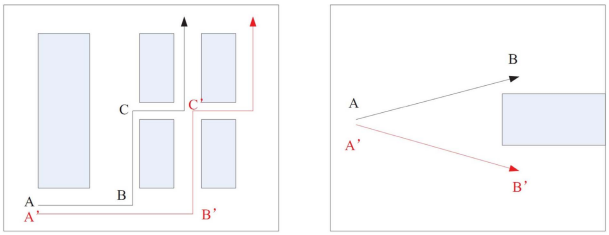


Fig. 2. Trace mismatching caused by motion sensors.

survey workload. In comparison, Slice utilizes the idea that the infrared light intensity decreases predictably with the distance, and maps the WiFi fingerprint to the floor plan effectively.

C. Machine Moving and Automatic Labeling

At a higher level of automation, it is possible to use a mobile machine to collect fingerprints. For example, PinLoc [16] employs a Roomba robot with a speed of 30 cm/s to collect physical layer information at each spot, which is distinguished by four virtual walls placed by users. The Simultaneous Localization and Mapping (SLAM) technique is able to localize a moving vehicle in an unknown environment without prior information, which depends on a variety of special and expensive hardware components, such as sonar [17], a millimeter-wave radar [18], and a laser range finder [19] to sense the environment.

There exist several sensors on smartphones that support linear positioning to varying degrees, including: motion sensors, camera sensors, and acoustic sensors (i.e., microphones). Unfortunately, all these sensors have serious limitations, yielding them inapplicable in our context. The low-end motion sensors on smartphones are rather inaccurate over long periods of time. For camera sensors, it is possible to determine a distance based on pictures [25], [26], provided a reference view. Quick movements will cause degradation in the estimate as the picture quality will degrade seriously when the sensor is moving. The ultrasonic based approach in [27] can accurately measure the distance of a smartphone to a sound source, but requires special modulation devices and multiple transmitters for sophisticated synchronization. To address this distance measurement problem, our paper exploits the idea that infrared light signal intensity is predictable with transmission distance, and presents an infrared light-based linear positioning system.

D. Other Positioning Methods

Indoor localization also benefit from many sensors, such accelerometers, light sensors, magnetic sensors, and cameras, especially on a smartphone platform. Many sensors are also joint together to acquire an accurate position, such as [28] discusses a hybrid indoor localization algorithm with pedestrian dead-reckoning, WiFi fingerprinting, and magnetic matching. PPNav [29] presents a peer-to-peer location system to track users' mobility with WiFi fingerprinting, magnetometer, and barometer on smartphones, and then guides other users to follow previous travellers' trace experience. iMoon [30] identifies user' position and moving direction with camera captured

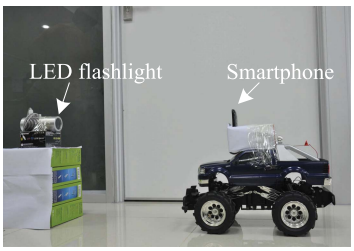


Fig. 3. Devices used in Slide.

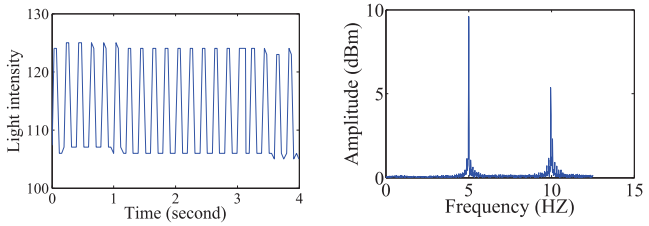


Fig. 4. Collected light samples by the light sensor on Samsung Omnia 2 and the results on the frequency domain.

photos and WiFi fingerprinting, to describe an 3D image model based indoor navigation system for smartphones. Instead of WiFi fingerprinting sampling, [31] shows a client/server based indoor positioning system, which gathers all WiFi APs' signal strengthes to build a dynamic WiFi radio map. Recently, visible light signal based positioning system has been presented. Rajagopal *et al.* [32] use camera sensors to detect the encoded position information in the LED lamps. AoA positioning algorithm and image analysis are also discussed in [33] and Luxapose [34]. In Epsilon [21] and LIPS [35], multiple transmitters are deployed to obtain more robust light signal. T-Tracking [36] takes advantages of sensor networks to detect the moving target by a novel design of "face prediction" instead of "target location prediction". In our paper, the light signal is considered as a location reference to speed up the fingerprint construction in WiFi based indoor localization, which is quite different from these visible light positioning.

III. Light-Based Linear Positioning

In our mobile fingerprinting method, Slide, the WiFi database is generally built with a mobile data collector (i.e., a moving vehicle with a sensing phone). It can collect fingerprints in a fast and accurate way.

Fig. 3 shows the components of Slide. A battery-powered LED flashlight is placed at the end of a straight line, and the remote control vehicle carries a smartphone and moves along the line. The smartphone determines its distance to the light source by measuring light intensity with its onboard light sensor while scanning the APs. The flashlight is switched on and off at a specified frequency (controlled by a built-in or external microcontroller); a remote-control toy vehicle[2] hosts a smartphone whose light sensor has its surface perpendicular with the central ray of the flashlight. While the smartphone is measuring light intensity, it records WiFi signal strengths at the same time.

Light-based linear positioning is based on the fact that the light intensity decreases predictably with the distance. Realizing this idea, however, has two challenges in a realistic environment. First, the ambient light from daylight or illuminating lamps may overwhelm the signal from the flashlight. Second, the vehicle may not always follow a strictly straight line, due to imprecision of control and vibrations, which results in significantly reduced light intensity and thus inaccurate distance estimation.

[2]Throughout the paper we consider a toy vehicle. In reality, a real vehicle can be used.

We solve the first problem using a flashlight with the ability to flash at a certain rate. Some commodity products (e.g., [37]) already provide this built-in feature, or this can be achieved with an external microcontroller. This allows us to separate the relevant light signal from the background in the frequency domain. For the second challenge, we employ a Nondeterministic Finite Automaton (NFA) model [38] to rectify the light intensity and obtain the distance with an improved accuracy. The following sections describe the details.

A. Interference Avoidance

The light intensity waveform on the light sensor contains a DC component imposed by the daylight, a 100 HZ component from the fluorescent lights as is typical indoor environments [39], and a component from the flashlight with a specified frequency. In our implementation, the flashlight has a flashing rate of 5 HZ and the smartphone samples the light sensor with 25 HZ. Fig. 4 shows a sequence of light signals detected by the Samsung Omnia 2 smartphone at a distance of 20 meters from the light source during a period of 4 seconds. The sensor's surface remains perpendicular to the central ray of the flashlight. Because the sampling period is 0.04 seconds, four times the period of the fluorescent light, the sampled fluorescent light strength is a constant in each each sampling slot. So, the daylight component and the 100 Hz fluorescent component can be removed by subtracting their summation.

After removing the DC component, Fast Fourier transform (FFT) is applied to transform the collected signal into the frequency domain. It is easy to identify spikes at 5 HZ and 10 HZ on the frequency domain, which correspond to the flashlight's signal. We use two band pass filters centered around 5HZ and 10HZ to extract these components, respectively, and then utilize inverse FFT to recover the light intensity of the flashlight.[3] On this basis, a log-distance path loss (LDPL) model is used to fit the relationship between the light intensity and the distance. In the following, when we say light signal, we refer to the signal of the flashlight.

Fig. 5 shows the prediction results by the LDPL model against real measurements. Each light intensity datum is the measurement from 10 seconds' data following the above method. It can be seen that the LDPL model fits the intensity-distance relationship very well. It should be noted that the experiment was conducted in a corridor where the floor is

[3]The frequency spike at 10Hz is a second harmonic wave signal of the infrared flashlight.

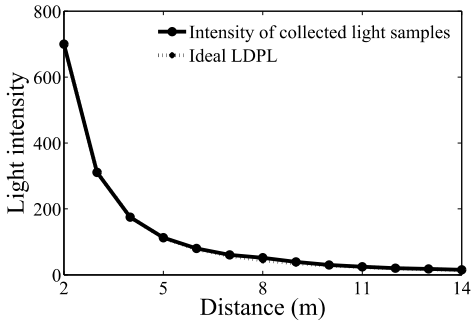


Fig. 5. LDPL model for the intensity-distance relationship.

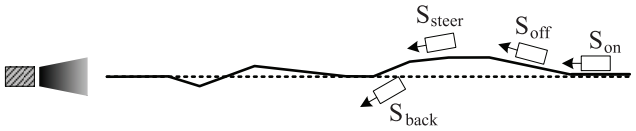


Fig. 6. Planned route (dashed line) and actual trace (solid line) by a remote control vehicle. The vehicle is in different states depending on its position and heading.

quite smooth and reflective. The reflective impact turns out to be insignificant on fitting accuracy, because the light sensor is embedded in the phone body, which has the ability to largely block the reflection from the floor.

B. Dealing With Mobility

Ideally, the vehicle should move straight along the central ray of the light source, so the distance of the light sensor can be accurately estimated. In practice, however, the inaccuracies in the remote control system may cause (temporary) deviation of the vehicle from the planned route, and the heading of the vehicle may also change due to vibrations. These factors, especially the latter one, would make the light intensity significantly smaller than the sensor should have received in a normal heading, thus resulting in errors in the distance estimation.

We deal with the issue by using the NFA model. The sequence of light samples is divided into intervals of 0.5 seconds each. For each interval, FFT and inverse FFT are applied to the data to compute a distance.[4] The middle of this interval is taken as the time associated with that distance. The instantaneous speed of the vehicle, v_i, at time t_i, is obtained by calculating the moved distance between two consecutive intervals divided by 0.5 seconds. Each interval also corresponds to a discrete point of time in the NFA.

Assuming that the vehicle is moving toward the light source (see Fig. 6), the NFA model comprises four states as follows:

1) S_{on}: on-track state. In this state, the vehicle moves along the straight line toward the light source and the vehicle's heading remains strictly forward. From the LDPL model, the light intensity will be increasing in this state.

[4]In each 0.5 second interval, the data length of FFT and inverse FFT are quite short, due to the low light sampling frequency 25 Hz. So, the energy consumptions of these signal processes are almost negligible in a smartphone platform.

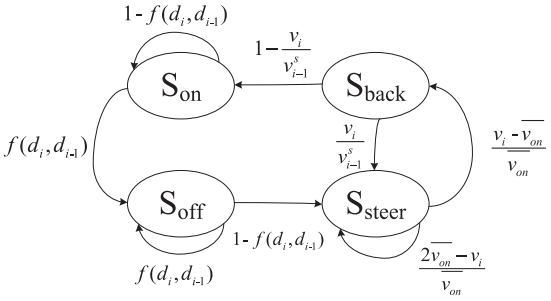


Fig. 7. State transition conditions and probabilities in NFA.

2) S_{off}: off-track state. In this state, the vehicle is not on the straight line, and the heading may not be strictly forward. This factor overwhelms the reducing distance, and as a result, the light intensity keeps reducing. Thus, the calculated distance appears to be increasing, as if the vehicle were moving in the opposite direction (i.e., far away from the flashlight). In our experiments, the light intensity is continuously sampled, and statistical results show that the light intensity in S_{off} state is lower than the previous state by at least 5%.
3) S_{steer}: steer state. The vehicle is being controlled by the user to adjust its heading and return to the straight line. During this process, v_i is larger than average speed $\overline{v_{on}}$ in previous S_{on} states, because the rectification of heading adds an extra contributing factor to the decreasing distance, as if the vehicle were moving faster toward the light source than it really does. In our experiments, the average speed is continuously determined with the NFA model. Statistical results show that v_i is 1.5x larger than $\overline{v_{on}}$.
4) S_{back}: back-to-track state. This state is used to mark the time when the vehicle is back to track. If the vehicle is back to track, v_i will be lower than the calculated speed in previous S_{steer} state. Statistical results show that v_i is more than 40% lower than previous S_{steer} state.

Since the distance estimation during S_{off} can be very inaccurate, the light intensity samples between an S_{off} and the next S_{on} will be discarded, and the missing distance will be estimated using interpolation according to the beginning and ending distances of that period.

The transition conditions and probabilities shown in Fig. 7 are as follows: (1) The vehicle is in S_{on} initially. If at time t_i, the calculated distance d_i is larger than d_{i-1}, the vehicle enters S_{off}. Otherwise, the vehicle remains at S_{on}. Define the function

$$f(d_i, d_{i-1}) = \begin{cases} 1, & d_i > d_{i-1} \\ 0, & d_i \le d_{i-1} \end{cases}$$

Then the transition probability from S_{on} to S_{off} is $f(d_i, d_{i-1})$, while the vehicle remains at S_{on} with probability $1 - f(d_i, d_{i-1})$. (2) In S_{off}, if at time t_i, the calculated distance d_i is still larger than d_{i-1}, the vehicle remains at S_{off}. Otherwise, the vehicle transitions to S_{steer}. That is to say, the vehicle stays at S_{off} with probability $f(d_i, d_{i-1})$ and transitions to S_{steer} with probability $1 - f(d_i, d_{i-1})$. (3) In S_{steer},

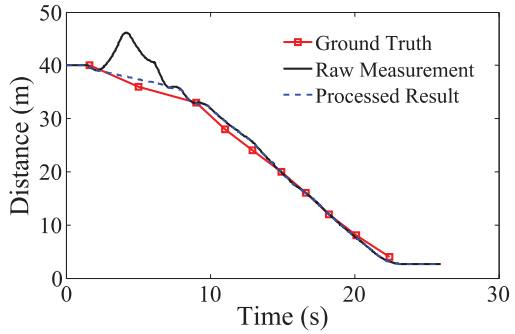


Fig. 8. Distance measurement and estimation.

the vehicle has two possible next states. It transitions to S_{back} with probability $\frac{v_i - \overline{v_{on}}}{\overline{v_{on}}}$ at time t_i, where v_i is the current speed and $\overline{v_{on}}$ the averaged speed on all previous S_{on} states. The vehicle remains at S_{steer} with probability $\frac{2\overline{v_{on}} - v_i}{\overline{v_{on}}}$. (4) In S_{back}, the vehicle transitions to S_{steer} with probability $\frac{v_i}{v^s_{i-1}}$, and to S_{on} with probability $1 - \frac{v_i}{v^s_{i-1}}$, where v^s_{i-1} is the speed calculated in the previous S_{steer}.

So far we have considered the case where the vehicle is moving toward the light source. The case of the vehicle moving away from the light source can be dealt with in a similar way, except that the sequence of samples need to be replayed in reverse order (note that all the calculations can be done off-line).

Fig. 8 shows the distance estimation results on a 40-meter line. The ground truth is obtained by fixing 10 marks on the line and manually recording the times when the vehicle passes the marks. The solid curve shows the calculated distance from the raw light intensity samples, while the dashed curve represents the results from the NFA model. It can be seen that while the raw data based estimation can generate an error up to 10 meters, the NFA model is able to keep the error within 1.4 meters, indicating that our method is tolerant of inaccuracies in vehicular control and mechanical movement.

IV. Fingerprint Misalignment Problem

In this section we describe the fingerprint misalignment problem, as well as a channel-based scanning method that can mitigate it.

A. Fingerprint Misalignment

The standard API on Android/Linux returns the RSS of all APs in range, after the network device performs a scan over all the channels. In the 802.11 standard, there are two modes of scanning: active scan and passive scan. Because active scan requires less time, as depicted in [40] and [41], most commodity device drivers use active scan to gather channel information.

Fig. 9 depicts an active scan process in 802.11. First, the device broadcasts a probe packet on a channel, and waits a certain amount of time to receive the response from APs in range. The device demodulates the received packets and calculates the signal-noise-ratio (SNR) later, and then switches to the next channel and performs the same routine. On a typical smartphone, an active scan takes around one second [13].

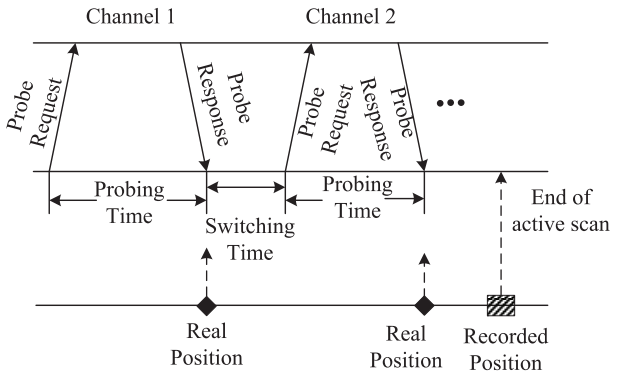


Fig. 9. Active scan in 802.11 and the fingerprint misalignment problem.

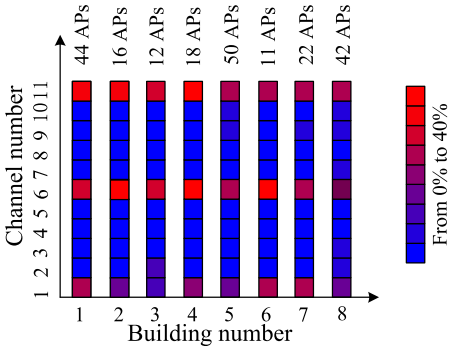


Fig. 10. AP distribution over 802.11 channels in 8 different buildings. Colors from blue to red represent increasing proportion.

In the standard implementation of an 802.11 driver, the scan results are returned to the user space only at the end of a full scan process. This would bring about some problem for mobile fingerprinting, as the vehicle may have already moved a certain distance during the scan process. That is, the RSS measured for many APs does not correspond to the position measured at the end of a scan process, resulting in *fingerprint misalignment*. For instance in Fig. 9, the square represents the recorded position of the device to be taken as the fingerprint position, while the diamonds represent the real positions corresponding to the collected RSSs. To enable qualitative comparison between different schemes, we roughly define the *misalignment error* as the duration of a scan multiplied by the average moving speed of the vehicle, denoted by v. Thus given the one-second scan delay, a mobile fingerprinting scheme would have a misalignment error of v.

B. Channel-Based Scanning

In order to get more prompt response from the device, we modify the active scan process in the device driver. Specifically, the driver sends a probe packet on a channel, waits certain time, and returns the received RSS information to the user. Then it switches to the next channel to repeat this process. The scan delay is reduced from about one second to $1/11 \approx 0.1$ seconds, thus the misalignment is reduced to $0.1v$.

In practice, most APs are by default configured to operate on a subset of the channels, such as channels 1, 6 and 11, for minimum interference. We have conducted a measurement study in 8 different buildings in two cities. Fig. 10 shows the

AP distribution over the 11 802.11b channels, which suggests that most of the time, the device needs to scan only a small number of channels. In our implementation, the distribution of APs on each channel is maintained by having the driver execute a full scan through all the 11 channels periodically (e.g., every 100 channel scans). Subsequently the driver only scans non-empty channels that contains at least one AP.

The channel based scanning on several android based commercial smartphone was implemented. Take Goolge_Nexus-S phone as an example. The phone uses a Broadcom network interface card and runs Linux Kernel 2.6.35-gingerbread. We made a few changes to *dhd_common.c* and *wl_iw.c* under *drivers/net/wireless/bcm4329*. The WiFi module was rebuilt and the kernel was reinstalled on the Android system, after which the Android API *wifiManager.getScanResults()* can return the results at a channel granularity.

The compatibility with the standard scanning routine is achieved by setting a control variable which can be controlled by the user-space applications. By default, the driver works in full scan mode, and switches to the channel-based scan mode upon detecting the change of the control variable.

Note that the basic idea of our paper is quite different from [40], which selects a subset of channels with a higher scanning frequency to reduce the scanning time. The main concern of our paper is the misalignment distance reduction, so the fingerprint location is captured after each channel scanning, rather than traditional all channel scanning.

C. Fingerprint Transformation

Due to the randomness in scan duration and variable vehicle speed, the sampled positions from each trip normally do not coincide. In order to combine the RSS samples of multiple trips, we choose reference spots on the traveled line (in our implementation with an interval of 1 meter), and approximate each sampled position with a nearest spot. By doing so, we are thus able to obtain an average RSS vector for each spot. The final fingerprint database is in the same form as the one generated by a manual fingerprint database, and therefore can be used by any fingerprint based positioning algorithm.

V. Slide in Practice

In this section we describe how Slide is used for two typical cases: straight paths and unobstructed indoor areas (open areas).

A. Slide for Straight Paths

We first consider a straight path, which can be fingerprinted by controlling a vehicle to move along the line in both directions. The main question is: how many trips are needed for the fingerprints to reach a stable state, given the constant fluctuation of WiFi signal.

Let us define a *fingerprint sample* as a vector $F = [s_1, s_2, \ldots, s_k]$, where s_i is the RSS of the ith AP. The *distance* between two samples F and F' is defined as $D(F, F') = \sqrt{\sum_{i=1}^{k}(s_i - s_i')^2}$. We conduct an experiment on a 40-meter line using a vehicle that travels at speed 4 m/s, and examine the stability of fingerprints for different numbers of trips (for mobile methods) or numbers of samples (for manual methods). On the line we mark 40 evenly spaced reference spots. For each of these spots we manually collect 20 RSS samples and take the average as the baseline.

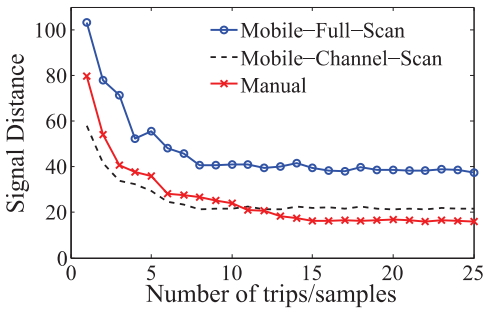


Fig. 11. Signal stability with number of trips/samples.

Three fingerprinting methods are compared: manual fingerprinting (Manual), mobile fingerprinting with full scan (Mobile-Full-Scan), and mobile fingerprinting with channel-based scan (Mobile-Channel-Scan). For each number of trips/samples, we compute the average RSS vector for each spot and obtain its distance to the baseline. Three types of smartphones, namely Google Nexus, Huawei U8825D, and Motorola XT882, are tested. They produce similar trends of convergence, thus we only report on the Google Nexus phone in the following.

Fig. 11 presents the stability results of the three methods against various numbers of trips/samples. As can be observed, Mobile-Channel-Scan reaches a stable point at 5 or 6 trips. By contrast, Manual does not converge until 15 samples. This leads to a huge difference in the time spent on fingerprinting. Roughly speaking, the 4 m/s vehicle takes a total of 60 seconds to complete the 6 trips along the 40-meter line, while Manual will need $40 \times 30 = 1200$ seconds to complete the same task, assuming a one-meter distance between reference spots and a duration of 30 seconds for the AP scans and human operations.

The vast difference between the efficiency of the two methods is mainly due to the faster sampling of the channel-based scan algorithm. In practice, most APs operate on channels 1, 6 and 11. The scanning algorithm used in Slide is able to focus on a small number of channels and avoid wasting time on empty ones, thereby providing more RSS samples within a unit time as well as allowing faster convergence of RSS vectors. In our experiments, all the APs are found on the three channels, whose scans take about 30% of the full scan time, leading to three times more samples being produced during the travel. That is why Slide takes only 5 trips, instead of 15, to converge on the fingerprints.

It is noted that only one light source is deployed in our system. If there are multiple straight line patterns, we can put the light source to the end of each straight line, and then measure the light intensity one by one. We have to emphasize that our system also works well for multiple light sources. In this case, their flashing rates are set to different frequencies (i.e., 3Hz, 5Hz, 7Hz, etc), so they have different light spikes in the frequency domain, which can be easily recognized with FFT and inverse FFT methods.

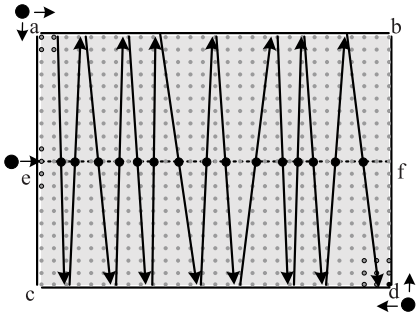


Fig. 12. Slide in a free space. Dots a, e, d are control points; other dots represent reference spots. The boundary lines are light-guided trips; internal arrow lines are accelerometer guided trips. All internal lines form a zigzag trip over the area, whose control can be accomplished at any control point.

B. Slide for 2D Free Spaces

Open areas such as empty spaces and atria are common in indoor environments. In these areas, users can move freely without following fixed directions. The lack of structural features causes difficulty for existing automatic fingerprinting schemes, and thus requires elaborately designed solutions.

A simple way of fingerprinting an open area is to divide the space into multiple parallel lines and then apply Slide to the lines separately. This *Slide-Parallel* approach, however, requires much effort in relocating the flashlight and manually orienting it at the end of each line, which we believe is an undesirable cost. We propose a more efficient scheme, called *Slide-Zigzag*, that involves less human effort. Fig. 12 shows an example area, where the lattice points represent the reference spots. We first fingerprint the four boundary lines, ab, cd, ac, bd of the area, which involves two *control points* at a and d, where the user needs to place the flashlight, have it properly oriented, and start controlling the vehicle. Afterwards, the user can stay at a control point (say a) and start controlling the vehicle to perform a zigzag trip over the area (see the internal arrow lines). The complete zigzag movement will be repeated five times, during which channel-based scanning is continuously executed to sample RSSs. Each time an RSS is obtained, it will be mapped to a nearest reference spot.

The key problem of the zigzag movement is how to obtain the vehicle's real-time position. Here we employ an accelerometer based dead reckoning method to calculate the position, with special care to curb the growth of error. It is known that a low-end accelerometer on a smartphone is highly unreliable over long periods of time due to quick error accumulation. However, we find that for short distances (e.g., within 10 meters), an accelerometer can give reasonably accurate results. Fig. 13 shows the error growth trends of several brands of smartphones running over a 40-meter line. The ground truth is again obtained from fixed marks and manual recording of the vehicles passing times. We can see that in general, the error grows super-linearly with time, exceeding 5 meters after a distance of 20 meters. Within 10 meters, the error stays within 3 meters, which suffices for fingerprinting.

To exploit the accelerometer's short-distance efficacy, we introduce some *auxiliary lines* into the area (see the dashed line ef). We first construct fingerprinting using the vehicles at the auxiliary lines (e.g., line e), and determine the end points of

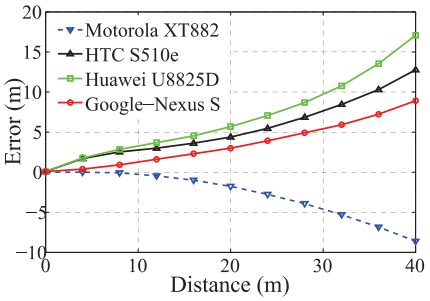


Fig. 13. Error growth trend with distance for different smartphones.

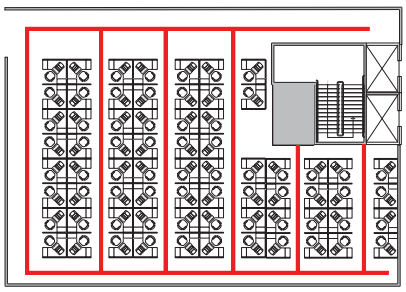


Fig. 14. The OFFICE scenario, with a size $20m \times 15m$. The red lines represent fingerprinted areas.

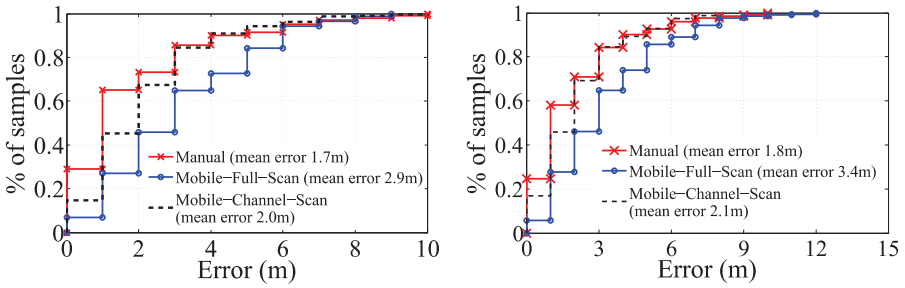
these lines using previously collected fingerprints on boundary lines ac and bd. During the zigzag movement, the vehicle tries to detect whether it hits an auxiliary line or a horizontal boundary line by matching its currently collected fingerprint against previously established ones. If so, it resets its current position to the matched one.

A zigzag trip can be easily followed with a modern toy car. As long as the vehicle's real-time position can be estimated, a valid fingerprint can be established and added to the database. In case a lattice point does not receive fingerprints, it takes the average of the fingerprints of its surrounding reference spots as its own.

VI. Experimental Evaluation

We have implemented the light-based linear positioning method on the Samsung Omnia 2 smartphone. The light sensor is an Intersil ISL29023 which can sense both visible and infrared light. To avoid visual discomfort we choose an infrared flashlight. The flashlight has a power rating of 5 watts and can be sensed by the phone at a maximum distance of 60 meters. It is controlled by an external microcontroller to flash at 5 Hz rate. Two remote-control toy vehicles are used in the experiments, namely SLOW and FAST, with average speed of 2 m/s and 4 m/s, repectively.

The experiments are carried out in three scenarios. The first, labeled CORRIDOR, is a 40 meters long straight corridor, by which we examine the performance of linear positioning and basic properties of mobile fingerprinting. The second scenario, OFFICE, as shown in Fig. 14, is an indoor office of area 300 m^2 with several paths, for evaluating Slide's performance in a 2D environment. The third scenario, EMPTY-SPACE, is an empty space of area 300 m^2. We hope to investigate how Slide works in a free space. Reference spots are marked evenly on the line or across the space in a lattice pattern with a one-meter interval.



(a) (b)

Fig. 15. Positioning error on a 40-meter line. (a) Slow vehicle with speed 2 m/s. (b) Fast vehicle with speed 4 m/s.

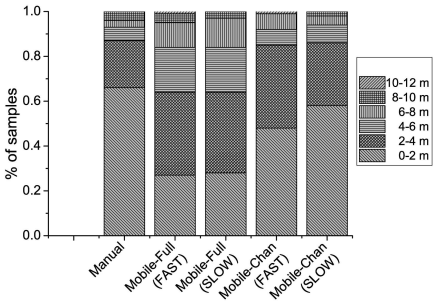


Fig. 16. Position error distributions on a 40-meter line.

We compare the performance of the aforementioned three fingerprinting methods. For Manual, we collect 15 samples of RSS vector for every reference spot and take the average as the final fingerprint. For Mobile-Full-Scan, 15 trips (the same number chosen in Manual) are performed to obtain stable fingerprints for each path. For Mobile-Channel-Scan, 6 trips are repeated to obtain the fingerprints for each path. To evaluate the system performance under different environmental factors, we start these trips in different time (8:00 am, 12:00 pm, 6:00 pm) of different days. All these methods output a fingerprint database in the same form; that is, every reference spot is assigned a fingerprint. We use the basic kNN algorithm [4] ($k = 1$ for CORRIDOR; $k = 3$ for OFFICE and EMPTY-SPACE) for calculating positions. Note that Slide is orthogonal to current WiFi localization algorithms in the positioning phase, so we use the basic kNN algorithm as an example. We believe that other methods will benefit from Slide as well.

A. Positioning Accuracy

The CORRIDOR scenario. We randomly sample 30 spots on the line for testing. Fig. 15 shows the cumulative distribution function (CDF) of positioning error for the two vehicles. For Manual, most of the sample spots have an error below 3 meters, with an average of 1.7 meters. By contrast, Mobile-Full-Scan performs much worse, due to the fingerprint misalignment error: less than half of the tested spots are positioned within 3 meter of their real positions. Furthermore, nearly 20% of all the sample spots are as far as 6 meters away from the ground truth. The average error is 2.9 meters for the SLOW vehicle, and 3.4 meters for the FAST vehicle, indicating that higher speed results in larger fingerprint misalignment, which further leads to larger positioning errors. Our method, Mobile-Channel-Scan, proves to be very helpful for positioning. On average, the sample spots have positioning accuracy to 2.0 meters and 2.1 meters for the SLOW and FAST vehicles, respectively. Fig. 16 presents the position error distribution with these fingerprinting methods. We can infer that Mobile-Channel-Scan performs better than Mobile-Full-Scan in most cases.

The OFFICE scenario. Fig. 17 shows the positioning errors of the different methods for the OFFICE scenario. Manual produces an average error of 2.3 meters. Mobile-Full-Scan produces average errors 2.8 meters and 4.2 meters for the SLOW and FAST vehicles, respectively, reconfirming the impact of fingerprint misalignment problem. Mobile-Channel-Scan can substantially reduce this impact, producing average errors of 2.3 meters and 2.4 meters for the same cases, which are very close to Manual's performance.

The EMPTY-SPACE scenario. The scenario is an empty space of size $15m \times 20m$. We first fingerprint the four boundary lines, and then choose an auxiliary line in the middle of the area (see Fig. 12). The zigzag movement is remotely controlled by a user at the top-left spot, and is repeated 6 times. Fig. 18 depicts the error statistics for Manual, Mobile-Channel-Scan and Mobile-Full-Scan. Manual and Mobile-Channel-Scan generate errors below 3 meters in most cases, with an average 2.3 and 2.7 meters, respectively, while Mobile-Full-Scan performs worse, with an average error of 4.6 meters.

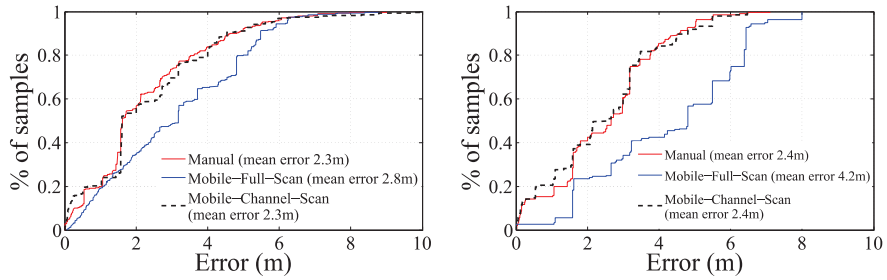
B. Fingerprinting Time

Assume there are N_{spots} reference spots in a field, and the vehicle travels at an average speed v. The time for manual fingerprinting can also be divided into two parts: identifying reference spots on the ground (e.g., by using a ruler) and manually linking a WiFi fingerprint to the corresponding spot. We use $T_{tagging}$, typically around 30 seconds, to represent the time for a user to wait on each spot and then move on to the next spot, and further assume a favorable case for this approach, where there are pre-existing position marks on the ground. Therefore,

$$T_{manual} \approx T_{tagging} \cdot N_{spots} = 30N_{spots}. \quad (1)$$

The total fingerprinting time of Slide comprises two parts: the time for setting up the flashlight and the time for the vehicle to traverse planned paths, that is

$$T_{slide} = N_{control} \cdot T_{relocate} + \frac{6N_{spots}}{v}, \quad (2)$$



(a) (b)

Fig. 17. Position error in the OFFICE scenario. (a) Slow vehicle with speed 2 m/s. (b) Fast vehicle with speed 4 m/s.

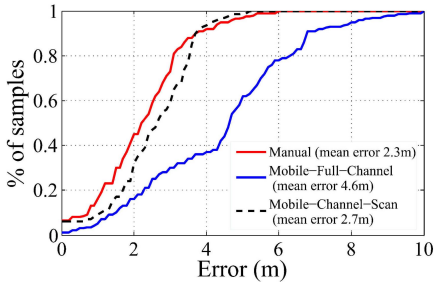


Fig. 18. Positioning error in the empty-hall scenario.

where $N_{control}$ is the number of control points, and $T_{relocate}$ represents the time for the user moving to a control point and placing the flashlight.

For a rough estimation, assuming the average path length is L in terms of number of reference spots, then $N_{control}$ is approximated with the number of paths to be fingerprinted, that is N_{spots}/L. In our experiments, $T_{relocate}$ is less than 60 seconds. Assuming a constant $T_{relocate} = 60$, then

$$T_{slide} = (\frac{60}{L} + \frac{6}{v})N_{spots}. \quad (3)$$

In a free space, there are some marks on the floor to complete a zigzag trip, which will bring in an additional manual fingerprinting time. In this case, the total fingerprinting time of Slide is

$$T_{slide} = (\frac{60}{L} + \frac{6}{v})N_{spots} + 30K_{spots}. \quad (4)$$

where K_{spots} is number of marks in a zigzag trip.

Obviously, the longer the average path length L, the smaller the *per-spot fingerprinting time*, defined as T_{slide}/N_{spots}. This suggests that Slide is particularly suitable for a large indoor environment.

The fingerprinting process of Mobile-Full-Scan is similar to Slide, except for the number of trips for each path. As illustrated in Section V, Mobile-Full-Scan takes 15 trips, instead of 6 trips in Slide. So, the total fingerprinting time of Mobile-Full-Scan is

$$T_{slide} = (\frac{60}{L} + \frac{15}{v})N_{spots}, \quad (5)$$

In a free space, the total fingerprinting time of Mobile-Full-Scan is

$$T_{slide} = (\frac{60}{L} + \frac{15}{v})N_{spots} + 30K_{spots}. \quad (6)$$

For an intuitive understanding, we evaluate the fingerprinting time saving in all scenarios. In the CORRIDOR scenario, the path length is 40 meter, and there are 40 reference spots. Thus, $T_{manual} = 30 \times 40 = 1200$ seconds. In Mobile-Full-Scan, with a vehicle traveling at speed 4 m/s and no extra zigzag marks, $T_{full} = (\frac{60}{40} + \frac{15}{4}) \times 40 = 210$ seconds. In Slide, $T_{slide} = (\frac{60}{40} + \frac{6}{4}) \times 40 = 120$ seconds, saving 90% and 43% of time compared with Manual and Mobile-Full-Scan.

In the OFFICE scenario, the path length is 15 meter, with 120 reference spots. According to Eq. (5) and Eq. (3), $T_{full} = (\frac{60}{15} + \frac{15}{4}) \times 120 = 930$ seconds, and $T_{slide} = (\frac{60}{15} + \frac{6}{4}) \times 120 = 660$ seconds. By contrast, $T_{manual} = 30 \times 120 = 3600$ seconds. Thus, Slides saves 82% and 29% fingerprinting time than Manual and Mobile-Full-Scan.

In the EMPTY-HALL scenario with an empty space of size $15m \times 20m$, SLIDE-ZIGZAG (an enhancement for empty spaces) involves only 5 control points and the total travel distance of the vehicle during one pass is 230m, so $T_{slide} = (\frac{60}{230} + \frac{6}{4}) \times 20 \times 15 + 30 \times 5 = 678$ seconds. In comparison, $T_{full} = (\frac{60}{230} + \frac{15}{4}) \times 20 \times 15 + 30 \times 5 = 1353$ seconds, and $T_{manual} = 30 \times 20 \times 15 = 9000$ seconds. Therefore, Slides saves 93% and 50% of time over Manual and Mobile-Full-Scan.

VII. CONCLUSION

This paper has presented a fast and accurate fingerprinting method, Slide, for WiFi-based indoor positioning systems, with an emphasis on the TSP problem. Different from previous methods that focus on fully automatic solutions for TSP, Slide provides a solution for semi-automatic TSP, trading off a part of automation for higher efficiency and wider applicability. Experiments show that Slide is significantly faster than existing methods (especially for public places with large free spaces), and can attain similar positioning accuracy to that of the manual fingerprinting method.

REFERENCES

[1] I. Sabek, M. Youssef, and A. V. Vasilakos, "ACE: An accurate and efficient multi-entity device-free WLAN localization system," *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 261–273, Feb. 2015.

[2] W. Kang and Y. Han, "SmartPDR: Smartphone-based pedestrian dead reckoning for indoor localization," *IEEE Sensors J.*, vol. 15, no. 5, pp. 2906–2916, May 2015.

[3] Z. Yang, C. Wu, Z. Zhou, X. Zhang, X. Wang, and Y. Liu, "Mobility increases localizability: A survey on wireless indoor localization using inertial sensors," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 54:1–54:34, 2015.

[4] P. Bahl and V. N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. IEEE INFOCOM*, Mar. 2000, pp. 775–784.
[5] 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.
[6] M. Azizyan, I. Constandache, and R. R. Choudhury, "SurroundSense: Mobile phone localization via ambience fingerprinting," in *Proc. ACM MobiCom*, 2009, pp. 1–12.
[7] L.-H. Chen, E. H.-K. Wu, M.-H. Jin, and G.-H. Chen, "Intelligent fusion of Wi-Fi and inertial sensor-based positioning systems for indoor pedestrian navigation," *IEEE Sensors J.*, vol. 14, no. 11, pp. 4034–4042, Nov. 2014.
[8] M. Youssef and A. Agrawala, "The horus location determination system," *Wireless Netw.*, vol. 14, no. 3, pp. 357–374, 2008.
[9] C. Wu, Z. Yang, Y. Liu, and W. Xi, "WILL: Wireless indoor localization without site survey," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 4, pp. 839–848, Apr. 2013.
[10] 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.
[11] 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.
[12] H. Wang, S. Sen, A. Elgohary, M. Farid, M. Youssef, and R. R. Choudhury, "No need to war-drive: Unsupervised indoor localization," in *Proc. ACM MobiSys*, 2012, pp. 197–210.
[13] G. Shen, Z. Chen, P. Zhang, T. Moscibroda, and Y. Zhang, "Walkie-Markie: Indoor pathway mapping made easy," in *Proc. USENIX NSDI*, 2013, pp. 85–98.
[14] Q. Jiang, Y. Ma, K. Liu, and Z. Dou, "A probabilistic radio map construction scheme for crowdsourcing-based fingerprinting localization," *IEEE Sensors J.*, vol. 16, no. 10, pp. 3764–3774, May 2016.
[15] *Heathrow Airport Terminal 3 Maps, Arrivals—Ground Floor*. Accessed: Jun. 27, 2017. [Onlile]. Available: http://www.heathrow-airport-guide.co.uk/terminal-3-map.html
[16] S. Sen, B. Radunovic, R. R. Choudhury, and T. Minka, "You are facing the Mona Lisa: Spot localization using PHY layer information," in *Proc. ACM MobiSys*, 2012, pp. 183–196.
[17] J. J. Leonard and H. F. Durrant-Whyte, "Simultaneous map building and localization for an autonomous mobile robot," in *Proc. IEEE IROS*, Nov. 1991, pp. 1442–1447.
[18] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, "A solution to the simultaneous localization and map building (SLAM) problem," *IEEE Trans. Robot. Autom.*, vol. 17, no. 3, pp. 229–241, Jun. 2001.
[19] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proc. AAAI*, 2002, pp. 593–598.
[20] J.-G. Park *et al.*, "Growing an organic indoor location system," in *Proc. ACM MobiSys*, 2010, pp. 271–284.
[21] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *Proc. ACM UbiComp*, 2012, pp. 421–430.
[22] P. Robertson, M. Angermann, and B. Krach, "Simultaneous localization and mapping for pedestrians using only foot-mounted inertial sensors," in *Proc. ACM UbiComp*, 2009, pp. 93–96.
[23] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proc. ACM UbiComp*, 2008, pp. 114–123.
[24] H. Abdelnasser *et al.*, "SemanticSLAM: Using environment landmarks for unsupervised indoor localization," *IEEE Trans. Mobile Comput.*, vol. 15, no. 7, pp. 1770–1782, Jul. 2016.
[25] J. G. Manweiler, P. Jain, and R. R. Choudhury, "Satellites in our pockets: An object positioning system using smartphones," in *Proc. ACM MobiSys*, 2012, pp. 211–224.
[26] C. Shi, J. Zhang, and Y. Zhang, "A novel vision-based adaptive scanning for the compression of remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 3, pp. 1336–1348, Mar. 2016.
[27] P. Lazik and A. Rowe, "Indoor pseudo-ranging of mobile devices using ultrasonic chirps," in *Proc. ACM SenSys*, 2012, pp. 99–112.
[28] Y. Li, Y. Zhuang, H. Lan, Q. Zhou, X. Niu, and N. El-Sheimy, "A hybrid WiFi/magnetic matching/PDR approach for indoor navigation with smartphone sensors," *IEEE Commun. Lett.*, vol. 20, no. 1, pp. 169–172, Jan. 2016.
[29] Z. Yin, C. Wu, Z. Yang, and Y. Liu, "Peer-to-peer indoor navigation using smartphones," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 5, pp. 1141–1153, May 2017.
[30] J. Dong, Y. Xiao, M. Noreikis, Z. Ou, and A. Ylä-Jääski, "iMoon: Using smartphones for image-based indoor navigation," in *Proc. ACM SenSys*, 2015, pp. 85–97.
[31] M. M. Atia, A. Noureldin, and M. J. Korenberg, "Dynamic online-calibrated radio maps for indoor positioning in wireless local area networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 9, pp. 1774–1787, Sep. 2013.
[32] N. Rajagopal, P. Lazik, and A. Rowe, "Visual light landmarks for mobile devices," in *Proc. ACM/IEEE IPSN*, Apr. 2014, pp. 249–260.
[33] G. B. Prince and T. D. C. Little, "A two phase hybrid RSS/AoA algorithm for indoor device localization using visible light," in *Proc. IEEE GLOBECOM*, Dec. 2012, pp. 3347–3352.
[34] Y.-S. Kuo, P. Pannuto, K.-J. Hsiao, and P. Dutta, "Luxapose: Indoor positioning with mobile phones and visible light," in *Proc. 20th Annu. Int. Conf. Mobile Comput. Netw.*, 2014, pp. 447–458.
[35] B. Xie *et al.*, "LIPS: A light intensity–based positioning system for indoor environments," *ACM Trans. Sensor Netw.*, vol. 12, no. 4, pp. 28:1–28:27, 2016.
[36] M. Z. A. Bhuiyan, G. Wang, and A. V. Vasilakos, "Local area prediction-based mobile target tracking in wireless sensor networks," *IEEE Trans. Comput.*, vol. 64, no. 7, pp. 1968–1982, Jul. 2015.
[37] *Tank007 PT10 LED Flashlight*. Accessed: Jun. 27, 2017. [Online]. Available: https://www.alibaba.com/product-detail/Professional-Military-Quality-Tactical-LED-flashlight_621654118.html
[38] M. O. Rabin and D. Scott, "Finite automata and their decision problems," *IBM J. Res. Develop.*, vol. 3, no. 2, pp. 114–125, 1959.
[39] Z. Li, W. Chen, C. Li, M. Li, X.-Y. Li, and Y. Liu, "FLIGHT: Clock calibration using fluorescent lighting," in *Proc. ACM MobiCom*, 2012, pp. 329–340.
[40] M. Youssef, L. Shahamat, M. Kleene, and A. Agrawala, "The IEEE 802.11 active probing analysis and enhancements," in *Proc. Int. Conf. Wireless Netw., Commun. Mobile Comput.*, Jun. 2005, pp. 1539–1544.
[41] I. Ramani and S. Savage, "SyncScan: Practical fast handoff for 802.11 infrastructure networks," in *Proc. IEEE INFOCOM*, Mar. 2005, pp. 675–684.

Kongyang Chen received the B.S. and M.E. degrees from Central South University, China. After that, he joined the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China. He is currently working toward the Ph.D. degree at the Shenzhen College of Advanced Technology, University of Chinese Academy of Sciences. His research interests include wireless sensor network and mobile sensing.

Chen Wang (S'10–M'13) received the B.S. and Ph.D. degrees from the Department of Automation, Wuhan University, China, in 2008 and 2013, respectively. From 2013 to 2017, he was a Post-Doctoral Research Fellow with the Networked and Communication Systems Research Laboratory, Huazhong University of Science and Technology, China. Thereafter, he joined the faculty of the Huazhong University of Science and Technology, where he is currently an Associate Professor. His research interests are in the broad areas of wireless networking, Internet of Things, and mobile computing.

Zhimeng Yin received the B.S. and M.E. degrees from the Huazhong University of Science and Technology, China. He is currently pursuing the Ph.D. degree with the University of Minnesota. His research interests include wireless networks and Internet of Things.

Hongbo Jiang (SM'14) received the B.S. and M.S. degrees from the Huazhong University of Science and Technology, China, and the Ph.D. degree from Case Western Reserve University in 2008. He joined the faculty of the Huazhong University of Science and Technology, where he is currently a Full Professor and the Dean of the Department of Communication Engineering. His research concerns computer networking, especially algorithms and protocols for wireless and mobile networks. He is an Editor for the IEEE/ACM TRANSACTIONS ON NETWORKING, an Associate Editor for the IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Technical Editor for the *IEEE Communications Magazine*.

Guang Tan (M'12) received the Ph.D. degree in computer science from The University of Warwick, U.K., in 2007. From 2007 to 2010, he was a Post-Doctoral Researcher at INRIA-Rennes, France. He is currently a Professor at the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China, where he works on the design of distributed systems and networks. His research interests are in the areas of peer-to-peer computing, wireless sensor networks, and mobile computing. He is a member of ACM and CCF.