QID: Identifying Mobile Devices via Wireless Charging Fingerprints

Deliang Yang*, Guoliang Xing†, Jun Huang‡, Xiangmao Chang§, Xiaofan Jiang¶
* Michigan State University, yangdeli@msu.edu
† The Chinese University of Hong Kong, glxing@ie.cuhk.edu.hk
‡ Peking University, jun.huang@pku.edu.cn
§ Nanjing University of Aeronautics and Astronautics, xiangmaoch@nuaa.edu.cn
¶ Columbia University, jiang@ee.columbia.edu

***Abstract*—Recent years have witnessed the increasing penetration of wireless charging base stations in the workplace and public areas, such as airports and cafeteria. Such an emerging wireless charging infrastructure has presented opportunities for new indoor localization and identification services for mobile users. In this paper, we present QID, the first system that can identify a Qi-compliant mobile device during wireless charging in real-time. QID extracts features from the clock oscillator and control scheme of the power receiver and employs light-weight algorithms to classify the device. QID adopts 2-dimensional motion unit to emulate a variety of multi-coil designs of Qi, which allows for fine-grained device fingerprinting. Our results show that QID achieves high recognition accuracy. With the prevalence of public wireless charging stations, our results also have important implications for mobile user privacy.**

***Index Terms*—Qi wireless charging, device recognition, real-time processing**

I. Introduction

Recent years have witnessed the increasing penetration of wireless charging base stations in public areas like offices, restaurants, and airports, etc. [1]. These is also a trend to embed wireless charging base stations in furnitures like desks and tables [2], [3]. It is estimated that nearly 600 million wireless charging devices were shipped during the year 2018 [4]. This emerging wireless charging infrastructure has presented new opportunities for precise user localization, where the base station learns the location and identification of the mobile device being charged. A number of different wireless- or ultrasonic-based approaches have been proposed for indoor localization [5]–[12]. Designed for providing continuous location of a moving user, they often incur significant overhead, e.g., due to the need of large-scale wardriving for collecting fine-grained signal fingerprints. In this work, we exploit wireless charging for a specific application scenario, where the user stays right next to the wireless charger, waiting for the phone to be charged. Therefore, the wireless charger localizes a mobile phone by simply referring to the already-known location of the registered charger.

Pervasive wireless charging stations provide high localization accuracy and high reliability at low deployment cost, which will enable a wide range of applications. For instance, a coffee shop may recognize its customers when they charge their phones on the coffee table, and provide customized services or location-based advertisements. For another example, when users charge their phones on the table instrumented with wireless charing during a meeting or lecture, the precise sitting positions of the users can be determined, which enables interesting interactions such as sharing documents in an ad-hoc group, sending instant messages, or exploring nearby people [13]. In addition to mobile device localization, the popularity of wireless charging infrastructure also provides opportunities for user authentication. For instance, a paid wireless charing service may use the charger to identify the phone and process the payment automatically.

To leverage the wireless charging infrastructure for user localization and identification, a key challenge is to reliably identify the wireless charging unit of mobile devices. Unfortunately, unlike network interfaces such as Wi-Fi and Bluetooth that have unique and fixed hardware addresses, the wireless charging unit of commercial off-the-shelf (COTS) mobile devices typically does not have a fixed hardware ID. For instance, according to the Qi standard [14], the identity of a power receiver is defined by a Basic Device ID, which can be a software-generated random sequence that may change each time the power receiver is booted.

In this paper, we present the design, implementation, and evaluation of QID – the first practical system that reliably identifies Qi-compliant mobile devices based on the hardware fingerprints. Specifically, QID augments standard-compliant wireless charging base station to extract features from the oscillator, coil, and controller of a Qi-compliant power receiver, while requiring no retrofitting or modification to existing Qi-compatible mobile devices. QID employs a 2-D motion controller to emulate the coil array in the Qi reference design (described in Section III) and regulate the inductive coupling between the power transmitting and receiving coils, which allows for fine-grained fingerprinting of the power receiver while optimizing the efficiency of power transfer. Experimental results based on 52 Qi-compatible devices show that QID achieves an overall identification accuracy of up to 89.7%, with an average of 85.3%. Our results also have important implications for user privacy. With the increasing prevalence of wireless charging stations in public areas, how to prevent the leakage of user's location opens up new research questions.

The rest of the paper is organized as follows. Section II

978-1-7281-6602-5/20/$31.00 ©2020 IEEE
DOI 10.1109/IoTDI49375.2020.00009

reviews related work. Section III introduces the background of the Qi specification. Section IV presents the challenges and the overview of the QID design. In Section V, we describe our QID sensor design for device fingerprint acquisition. Section VI presents the QID motion control design. Section VII discusses the feature selection and classification at the server side. The implementation and the evaluation results of the QID are discussed in Section VIII and Section IX. Finally, we conclude this paper and discuss the future work.

II. Related Work

Device identification has been studied for a wide range of communication systems. The existing techniques can be broadly classified into two categories. One category uses the fingerprints of RF signal introduced by hardware imperfection of frequency generator on the devices. The other category uses temporal features, i.e. the clock skew introduced by the minor difference in the oscillator among the devices. The clock skew mainly affects the time interval of the transmitted packets.

RF Signal Fingerprinting. PARADIS [15] identified the source network interface card (NIC) of an IEEE 802.11 frame through passive radio-frequency analysis. Specifically, it uses I/Q origin offset, frequency error, and SYNC correlation to distinguish the devices. Caraoke [16] separated devices by their carrier frequency offset differences to avoid wireless collisions in an e-toll transponder network. Similarly, Danev et al. [17] achieved wireless sensor recognition using radio frequency transient characteristics. Eletreby et al. proposed Choir [18], a system that disentangles collisions in LoRa LP-WAN by distinguishing the sensor nodes using their time, frequency and phase offsets caused by hardware imperfection. However, these techniques cannot be applied to wireless charging, because extracting the fingerprints of RF signal often requires expensive equipment. For example, [15] used Agilent 89641S vector signal analyzer to capture the error vectors in the IQ plane. Moveover, wireless charging adopts resonant coupling to transfer energy, where both the carrier frequency and amplitude are variable, which makes it impossible to infer the device identity using the RF signal in wireless charging.

Clock Skew Fingerprinting. Kohno et al. [19] used the TCP timestamp option to estimate a device's clock skew. Similarly, Cristea and Groza [20] studied how to fingerprint smartphones remotely via the Internet Control Message Protocol (ICMP) timestamp response. While these two studies focused on traffic and driver-level signatures, other systems explored hardware-level features to distinguish devices. Huang et al. [21] used temporal features of Bluetooth baseband embedded in the chipset firmware to fingerprint Bluetooth devices. However, one key difference between these scenarios and wireless charging is that the clock skew fingerprints of the mobile device is heavily dependent on the device placement. Moreover, the placement of the device on the charger pad is unpredictable, which casts significantly difficulty in building a precise model for each device. We will further discuss the challenges in detail in Section IV.

TABLE I
PRx timing constraints during the Qi power transfer phase.

Parameter	Symbol	Target (ms)	Max (ms)
CEP Interval	t_{interval}	250.0	350.0
RPP Interval	t_{received}	1500.0	4000.0

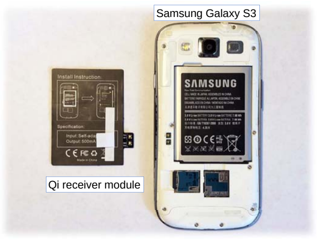


Fig. 1. An attachable Qi-compatible power receiver for Samsung Galaxy S3.

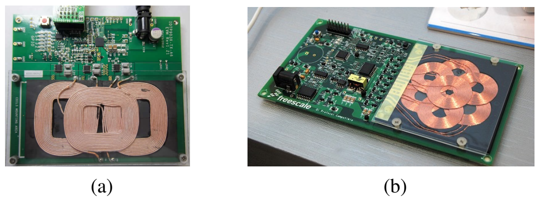


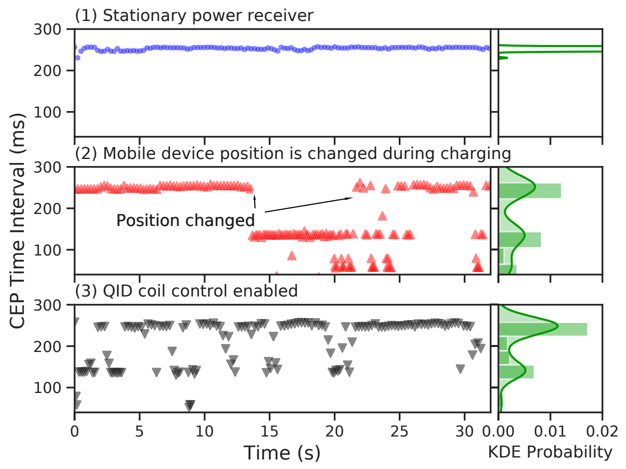
Fig. 2. Examples of the multi-coil PTx designs in Qi.

There exist other device identification technologies based on fingerprints of acoustic sensor [22], [23], and inertia sensor [24], [25]. Although these methods may achieve acceptable accuracy, they require reading the data or sensor samples from the phone directly, which can be intrusive.

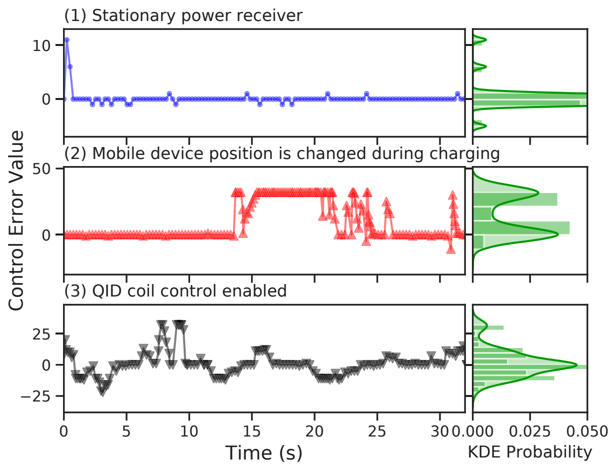
In addition to the device identification, Lu et al. [26] proposed a wireless charging network system, where multiple wireless chargers communicate with the server or adjacent wireless chargers to provide pay-per-use charging service. We note that reliable charging device identification is the basic building block for such applications.

III. Background

Qi is an open standard that defines wireless power transfer over short distances. A typical Qi system consists of a power transmitter (PTx) installed on a Qi base station and a power receiver (PRx) attached to or installed in a Qi-compliant mobile device. The PTx comprises a transmitting coil, called Primary Coil, which generates an oscillating magnetic field, that induces an alternating current in the receiving coil, namely the Secondary Coil, of the PRx. The PRx communicates with the PTx via backscatter modulation of the current draw, and primarily sends two types of messages to optimize the power transfer. The control error packet (CEP) carries an integer that indicates the difference between the desired power level and the received power level. The received power packet (RPP) reports the average level of power received in the past period. Throughout the process of charging, CEPs and RPPs are trans-



(a) The CEP time interval vs. sampling time in 3 independent experiments.



(b) The CEP value vs. sampling time in 3 independent experiments.

Fig. 3. Examples of feature acquisition in three configurations: (1) stationary PRx; (2) PRx position changed during charging; (3) QID is enabled.

mitted periodically. Table I shows the CEP and RPP intervals specified by the Qi standard. Based on the information in CEPs and RPPs, the PTx adjusts the carrier frequency and amplitude of the primary power signal to optimize the coupling between the coils of PTx and PRx. The combination of the carrier wave frequency and amplitude is defined as the operating point.

Qi specifies multiple reference receiver and coil designs [27]. Fig. 1 exemplifies one of the available designs and Samsung Galaxy S3 that supports attachable Qi-compatible PRx modules. It has a pair of terminals (+5V and GND) that connects to the output of the PRx. In such a design, the PRx is an independent module and does not communicate with the phone. The attachable PRx modules provide the wireless charging capacity to those devices that originally do not ship with the wireless power receivers. In this work, we assume each such module represents a user identity, since it is an independent component (either attached outside or pre-installed inside). It outputs stable current at 5 V for charging with its maximum capacity most of the time.

In addition to the PRx design, Qi also specifies more than 30 type A and 7 type B PTx designs, where type A designs have one or more Primary Coils but *only one* of them can be activated at a time, while type B designs support an array of Primary Coils and *one or more* Primary Coils can be activated to provide wireless power to multiple PRxs simultaneously. Fig. 2 [28] shows two examples of the multi-coil chargers. Compared to the single coil designs, multi-coil PTx enlarges the possible coupling area with the PRx, thus providing more flexibility in the device placement. As a result, the coil array PTx designs become more prevailing on the market.

Qi also supports a serial number of at least 20 bits, also known as the Basic Device ID. However, a PRx can also use a random number generator to dynamically change the Basic Device ID, so that every time the user puts the mobile device onto a PTx, the Basic Device ID updates. Such a random ID invalidates device ID based applications, which inspires us to design a system to identify a device using its hardware fingerprints.

IV. Design Challenges and System Overview

A. Design Challenges

In this work, we choose the PRx temporal and control scheme features of charging process to identify the mobile device, which include CEP time intervals and values. As discussed in Section III, these features can be easily extracted from any Qi-compliant devices, which ensures the compatibility and easy deployment of our system. However, the following challenges need to be addressed due to the intrinsic characteristics of wireless charging environment.

Noise in temporal features caused by power transfer. The wireless power transfer happens in the rapidly-changing high power electromagnetic field between the two coupling coils, casting more noise than typical RF wireless systems. For instance, as shown in Fig. 3a(1), the measured packet intervals have significant fluctuations. The standard deviation of the CEP time interval is more than 4.4 ms, corresponding to 1.7% error, which makes it difficult to distinguish between different charging devices.

Undesirable stable operating point. Qi wireless charging has a well-designed feedback control loop. The wireless power transfer process is usually stabilized at an operating point within hundreds of milliseconds (3 to 5 CEPs). Although this is a desired feature in terms of maintaining high charging efficiency, it brings a major challenge in recognizing the target device. Fig. 3a(1) and Fig. 3b(1) show the experimental result of such a scenario, where the phone remains stationary on the charging pad. As shown, the selected features, both the CEP time interval and CEP values, remain unchanged during the charging process, which eventually raises the recognition error rate.

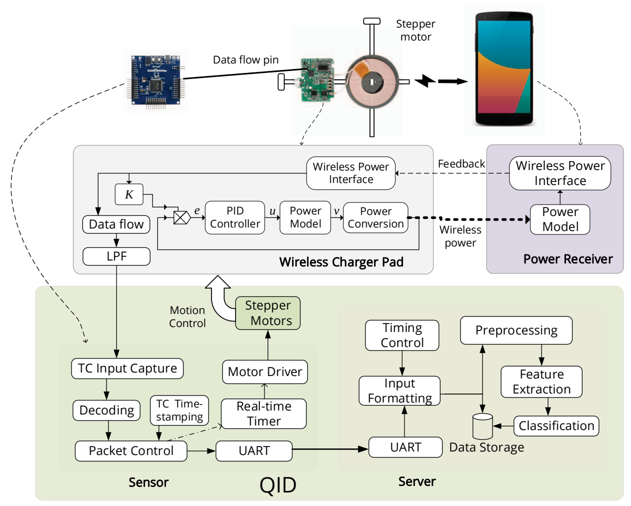


Fig. 4. The system architecture of QID.

Unconstrained device placement. The third challenge, the most difficult one in our case, is that the selected features are dependent on the phone placement. In other words, if the user alters the PRx position, the feature values can change dramatically. Fig. 3a(2) and Fig. 3b(2) demonstrate how the CEP time interval and control error value change with respect to the phone placement. During the measurement, after we rotate the phone with a random angle, the CEP time interval decreases from about 250 ms to 160 ms, and the Control Error value increases dramatically from 0 to 30. In a real-world scenario, the placement of a mobile phone is often unpredictable. As a result, the errors are accumulated over time, which eventually renders the recognition unsuccessful.

B. System Overview

We now provide an overview of the QID system. The system architecture is shown in Fig. 4. It consists of 3 components, namely an off-the-shelf Qi wireless charger, the QID sensor, and the QID server. The QID sensor is responsible for collecting a selected set of features from wireless charging and uploading the data to the server, while the QID server is responsible for the feature extraction and device classification. The QID server can connect to the QID sensor directly (e.g., through UART) or resides on the cloud and communicates with multiple QID sensors through the Internet, enabling tracking the target device at different charging locations.

The QID sensor can work with most Qi-compliant chargers. It does not modify any of the charger pad circuits. What QID sensor needs from a Qi-compliant charger is a *test pin* that outputs the filtered data bit flow. We note that such data pin is indispensable for the Qi charging system because the PTx requires the feedback from the PRx. Reading data flow from the pin does not affect the operation of the Qi charging system. Therefore, thanks to the minimal modification requirement, QID can be easily integrated with off-the-shelf Qi chargers. After connecting the test pin and mounting the charger coil to QID sensor, the platform is ready for device fingerprinting. The QID sensor consists of a motion control hardware component and a software component for feature collection. The design of the motion unit is discussed in Section VI. The motion unit hosts the charging pad and moves it according to certain pattern, within a range of 10 cm. This allows to fingerprint PRx dynamically at different relative positions between the PTx and PRx coils, resulting in higher identification accuracy. We note that the motion unit is connected to a separate control module, and it does not require wire connection with the charger itself. As a result, it can be integrated with any off-the-shelf charger easily.

The motivation of adopting the motion unit is two-fold. First, it can be easily integrated with single-coil chargers and improve the performance of classification accuracy as well as power delivery. Second, it can emulate many emerging new chargers with multi-coil Qi-compliant power transmitter design [28]–[30]. As described in Section III, each coil on such transmitter is controlled by an individual switch or a separate bridge. The PTx can select the optimal coil to deliver the wireless power to the PRx. Thus it enlarges the coupling area between the PTx and PRx and provides more flexibility in the device placement. Despite these advantages, it is difficult to exploit the multiple coils of Qi chargers for fingerprinting in practice, since there exists a large number of heterogeneous designs as specified by Qi [27]. To address this challenge, the QID sensor extends the design of the physical coil array to a *mobile coil* by equipping the primary transmitter coil with a motion unit. Such a design effectively emulates a variety of different multi-coil designs of Qi, while it tackles the design challenges mentioned in Section IV-A for the following reasons. First, the noise within the temporal features can be controlled and even filtered out in post-processing, because the fingerprints are collected from multiple coil locations, a more complete device profile can be built. Second, the wireless power transfer process to hop between different operating points when the charging coils are switched. Thus we can infer the PRx control scheme from the transient states between the operating points, which can be used to differentiate different Qi modules. Finally, the feature uncertainty caused by the phone placement can be essentially mitigated because the coil array covers a range of device positions on the charger pad.

In addition to the motion unit, the QID sensor also extracts and timestamps every packet in the data flow. A challenge in the design is to ensure the PRx is correctly located and measured. The details of the QID sensor are discussed in Section V. The last component, namely the QID server, reads all the data sent by the QID sensor. As the packet is in byte representation, the server needs to parse each field in the packet. Then, the server performs feature extraction and classification, which are discussed in Section VII.

V. Feature Selection and Acquisition

A. Selecting Hardware Fingerprints

To reliably identify Qi-compliant devices, QID leverages hardware fingerprints extracted from the following three PRx components. The selected fingerprints should be device-specific, time-invariant and discriminative. We notice that although QID is able to sense many of the analog signals, such as current magnitude, carrier wave frequency, and duty cycles, we are not going to employ them as recognition features. Because these analog features exhibit significant variance as the operating point changes, such that the feature values largely overlap between different devices, defying the success in the device identification.

Onboard oscillator. The PRx controller chip of Qi utilizes an internal oscillator to generate the clock signal. It is well known that oscillators have distinctive drifts due to factors like hardware manufacturing variations [19]–[21]. We thus exploit the drift of the PRx oscillator as a feature to identify the device under charging. For example, Panasonic AN32258A [31], a commercially available Qi receiver IC, utilizes an internal oscillator. NXP MWPR1516 [32] also uses internal Low Power Oscillator (LPO) as the clock source. We note that the Qi receiver ICs typically have low clock accuracy as they are not designed for data communication. For instance, the receiver IC NXP MWPR1516 has a clock accuracy tolerance of as high as $\pm5\%$; Rohm BD57011AGWL data sheet [33] also indicates that the driving frequency of the communication signal is between 1.92 and 2.08 kHz, which corresponds to a 4% frequency error tolerance. In comparison, the clock frequency tolerance is ±50 ppm for Bluetooth [34] and ±40 ppm for Zigbee [35]. Therefore, the clock drift effect of Qi is highly device dependent and much more significant than other wireless communication systems. Although drift variations like this can be used to differentiate different devices, it is difficult, if not impossible, to directly measure the clock drifts in COTS devices. Our key observation is that the Control Error Packet (CEP) time interval yields high variance among the devices around the target value specified in Qi (see TABLE I). Fig. 5 shows that the CEP time interval distribution of 42 devices spans a range of (238, 270) ms in the time domain. Therefore, the PRx oscillator can be inferred and fingerprinted by measuring the period drift of the control packets. However, some devices, for example, “A2” and “C2”, or “F6” and “F7”, yield close CEP time interval values.

Receiving coil. Different Qi-compliant devices may have different coil shapes, diameters, and layouts. Generally, a larger PRx coil has a larger contact area between the PRx and the PRx coils, leading to more flexibility in placing the device. In our scenario, the receiving coil diameter can be fingerprinted based on the area that the PTx interacts with the PRx. We discuss how to measure the contact range of the PRx coil in Section VI-B.

PRx controller. The Qi standard does not specify the exact period of control packet transmission. We observe that the periods of the CEPs do differ across devices of different manufacturers. Such vendor-dependent controller implementations can be exploited as a fingerprint to differentiate devices from different manufacturers. For example, Texas Instruments bq51013B [36] sends the CEPs with an interval of 240 ms, while Panasonic AN32258A sends the CEPs at a period of 160

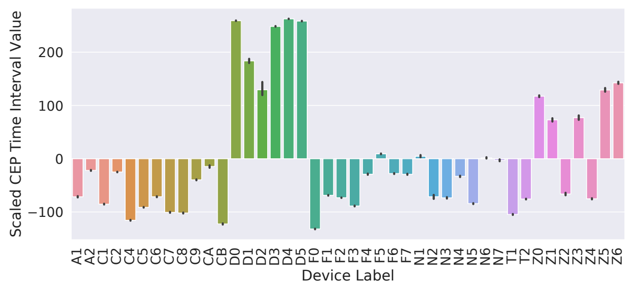


Fig. 5. The scaled and zero-meaned CEP time interval distribution of 42 evaluated devices. The first letters of the devices represent the brands of the receivers, while the following digit represents the specific label in its brand. The error bar shows the standard deviation of the CEP time interval. The corresponding actual time interval spans a range of (238, 270) ms.

ms. This feature is not related to clock error but a value chosen at design time by the manufacturer. Intuitively, determining the IC manufacturer improves the recognition accuracy by narrowing the categories of the devices. In addition to the packet period, the value carried in the CEP is also specified by the receiver IC manufacturers. For example, we observe that the maximum control error value sent by brand “C” devices is 30, while the brand “Z” devices can send the control error values as high as 80. Therefore, it is another feature that may distinguish the device brand.

B. Temporal Feature Acquisition

We now discuss how the QID sensor collects temporal features of Qi packets. To decode the bits, the QID sensor uses a timer to measure the width of each pulse. As the data sent by the PRx is encoded with a differential bi-phase scheme, we can convert the pulse widths to bit values. The decoded bits are then grouped into bytes.

A Qi packet consists of preamble, header, payload, and checksum. The QID sensor timestamps the packet after the 11th bit in the preamble phase of each packet. The corresponding packet time interval is then the difference between two consecutive timestamps. As described in Section III, the Qi protocol defines two types of packets that have fixed time intervals. QID sensor mainly observes and analyzes the CEP time interval to infer the PRx oscillator because the CEP is the most frequent type of packets that are sent during the wireless charging process.

VI. QID Motion Control

A. Motion Platform Design

In this subsection, we present the mechanical design to enable the movement of the charger coil, as illustrated in Fig. 6. It requires two linear slides powered by a stepper motor individually. First, the bottom slide is fixed on a surface. Next, the upper slide is placed with its axis direction perpendicular to that of the bottom one. The upper one is attached to the bottom one’s slider. Finally, the charger coil is attached to the slider of the upper linear slide. The two stepper motors are controlled independently to drive the coil in an X-Y plane

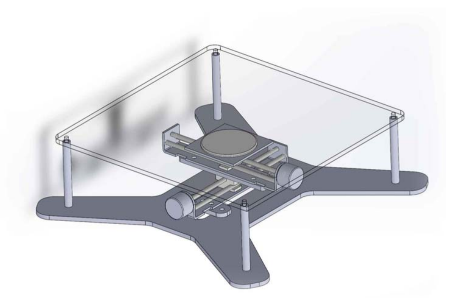
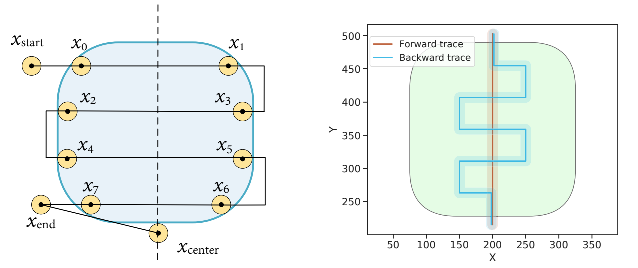


Fig. 6. Mechanical design - the charger pad is controlled by two stepper motor linear slides, moving in a 2-D surface.



(a) Symmetric axis searching. The upper left point is the starting point.

(b) The designed trajectory of the charger coil center within one complete fingerprinting scan.

Fig. 7. Trajectory design of the QID sensor.

to form a mobile coil. It thus allows for more flexibility in the PRx placement. The user can place the device with any location and any angle in the designated area. Correspondingly, we define the axes of the bottom slide and the upper slide as the x axis and y axis respectively. As the lengths of the two slides are the same, the working space of the PTx coil is defined as a square area. We envision that the QID motion platform can enable other applications, such as locating the PRx and searching for the optimal charging operating points, which are critical for optimizing the charging efficiency [14]. We leave these applications for future work.

B. QID Sensor Motion Control

In this subsection, we discuss the control schemes for the proposed QID sensor motion platform.

1) Contact area boundary detection: The detection of the PRx boundary allows for better coil movement control. For example, the stepper motor can adapt to a higher speed if the PRx is out of the contact range, or the trajectory can be optimized to avoid unnecessary moves, such that the total time needed for collecting sufficient features can be reduced. The QID sensor utilizes a timer to detect the contact boundary. Each time the QID sensor receives a new packet, it reads the real-time timer (RTT) to update a value t_{last}. In the meantime, the QID sensor reads the RTT with a period of 10 ms to fetch the current time t_n and compares it with t_{last}. Then the condition that the PRx is out of the contact range is given by:

$$t_n > t_{last} + T_{timeout},$$

where $T_{timeout}$ is the allowed time that the PRx does not send any feedback. We choose $T_{timeout}$ to be 350 ms, which is the maximum CEP time interval in the Qi standard, as presented in Table I. If the condition is met, the QID sensor determines that the PRx loses its contact with the PTx.

2) PRx symmetric axis alignment: Next we discuss how the QID sensor finds the symmetric axis of the PRx coil along the y axis. Finding the symmetric axis is important because it is the reference for the fingerprinting trajectory.

We assume that the device is in the contact range once the user puts it on the charger pad. The PRx symmetric axis alignment is achieved as follows. In the beginning, the PTx coil moves along the positive direction of the y axis until it is out of the contact range. Next it moves to the negative direction of both x and y axis, reaching the starting point shown in Fig. 7a (upper left corner). From there, the coil starts to move in an "S" pattern and sweeps across the PRx coil for four times, generating a sequence of the detected boundary $[x_0, x_1, ..., x_7]$. Then the x value of the symmetry axis is

$$x_{center} = \frac{1}{8}\sum_{i=0}^{7} x_i$$

Finally, the PTx coil is aligned to the x_{center} with its y coordinate value right out of the contact range boundary. This location is the starting point of the coil in the fingerprinting phase.

We note that, for a multi-coil PTx, this phase can be achieved by switching between the coils and identifying the one with the highest coupling.

3) Fingerprinting trajectory planning: Now we present the PTx coil trajectory design when the QID sensor collects fingerprints from the PRx. We take two factors into account when designing it. On the one hand, it is crucial to ensure the QID sensor captures adequate data from the PRx in the multicoil array, such that QID records the complete feature profile of the PRx. On the other hand, the more data points are measured, the more time it takes. Typically 4 or 5 packets can be collected during one second. If we plan to record 3,000 CEPs, it may take more than 10 minutes and exceed the time one leaves the phone on the charger pad. Therefore, we need to find a trade-off between the spatial data diversity and the measurement delay. In our design, we assume that the PRx will be left stationary on the platform for a time window of at least 90 seconds, such that the QID sensor captures adequate fingerprints for device identification.

The designed fingerprinting trajectory of the QID sensor is shown in Fig. 7b. It comprises two sessions, namely the forward one and the backward one. During both sessions, the QID sensor records all the packets sent by the PRx and uploads them to the server. The forward session starts from the end point of the PRx symmetric axis alignment phase. The PTx coil is driven to move along the positive direction of the y axis. Once it enters the contact range, the coordinate value y_{start} is recorded. In the meantime, the moving speed of the coil is set to a lower speed. It stops for 1 second each time after moving forward for about 8 mm (corresponding to 40 control units in Fig. 7b) to wait for the operating point hopping, which also mimics the coil switching in an array. Once the PTx loses contact with the PRx, i.e., the PRx is out of the contact range, the boundary point y_{end} is recorded, which also marks the end of the forward session. The backward trace is similar to the center alignment one, which is in "S" shape. The major difference is that the distance Δx between the farthermost point and the symmetric axis is about 10 mm along the x axis, corresponding to 50 control units. The movement along the negative y direction is divided into 6 sections. Each of them is

$$\Delta y = \frac{y_{end} - y_{start}}{6}$$

At each turning point, the PTx coil stops for 1 second to wait for the operating point hopping. After the coil reaches the forward session starting point (x_{center}, y_{start}), the fingerprinting phase finishes. We define a complete scan as the completion of both the forward and backward sessions, and the data collected during this phase are defined as a *scan sample* correspondingly. It is later post-processed at the QID server.

There are several reasons why such a trajectory is designed. First of all, it covers the central area of the contact range. Even if the PRx changes its placement angle next time, the central region still largely remains overlapping. As a result, the two independent scan samples do not deviate significantly. Second, the distance Δx is carefully chosen to accommodate different coil shapes and sizes. No matter how the PRx is placed, the planned trajectory falls within the contact range for most of the time. The trajectory is also able to tolerate the symmetric axis alignment error up to about 8 mm (corresponding to 40 control units). Finally, the same location is fingerprinted for at most once, as the forward and backward traces do not overlap except the region where the coil center enters or exits the contact area.

VII. Feature Extraction and Device Classification

In this section, we present how the QID server extracts the features from the measured data and classifies the features into different device classes.

A. Feature Extraction

We note that the contact range diameter of the receiving coil physical feature can be simply calculated as $y_{end} - y_{start}$. The oscillator features and the PRx control schemes require additional processing to obtain. We discuss them as follows.

1) CEP interval features: The QID server uses a local maximum searching algorithm to extract the CEP time intervals that represent the feature of PRx onboard oscillator. First, all the CEP time intervals are processed with a fixed bandwidth Gaussian kernel density estimator (KDE). Unlike the histogram, the Gaussian KDE represents the probability of each point in the feature space with a fixed-variance Gaussian distribution, i.e., the Gaussian kernel, and then the

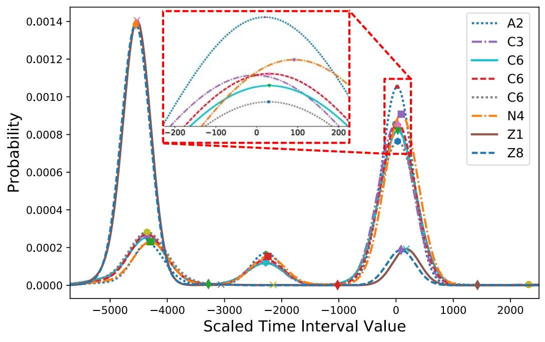


Fig. 8. Gaussian kernel density estimation of CEP time intervals. The letters indicate the device brands. The number indicates each unique device of its brand. The 0 in the horizontal axis corresponds to 240 ms in real time scale.

TABLE II
THE LIST OF FEATURES EXTRACTED FROM A COMPLETE SCAN.

Feature Group	Feature	Value Range
Oscillator feature	Domain 1 CEP interval	[40, 60] ms
	Domain 1 CEP interval peak log-probability	
	Domain 2 CEP interval	[140, 160] ms
	Domain 2 CEP interval peak log-probability	
	Domain 3 CEP interval	[235, 270] ms
	Domain 3 CEP interval peak log-probability	
Sample time	Time needed in a complete scan	>0
PRx controller feature	Number of packets	>0
	CEP value = 0 frequency	[0, 1]
	CEP value ∈ [1, 5] frequency	[0, 1]
	CEP value ∈ [6, 10] frequency	[0, 1]
	CEP value ∈ [11, 20] frequency	[0, 1]
	CEP value ∈ [21, 30) frequency	[0, 1]
	CEP value = 30 frequency	[0, 1]
	CEP value ∈ (30, 127] frequency	[0, 1]

estimation output is the averaged sum of all the individual kernel probability estimations. The output of the KDE is actually the probability density function (PDF) of CEP time interval.

$$d(t) = pdf(t), t \in [40, 270] \text{ ms}$$

Note that the variable t here is in time domain representation, while the feature values correspond to the QID sensor controller timer values.

Next, the QID server extracts the CEP time intervals that achieve local maximums in the PDF. We observe that the CEP time intervals typically fall into 3 domains, corresponding to [40, 60], [140, 160], [235, 270] ms in time domain (as shown as the three local maxima in Fig. 8). Therefore, it is feasible to find the peaks directly without calculating the derivative of the CEP time interval PDF. Specifically, the 3 domains are denoted as D_1, D_2, and D_3 respectively. Then the feature CEP time intervals and their corresponding log-probabilities are

$$t_{pi} = \arg\max_{t \in D_i} d(t), i = 1, 2, 3.$$

$$p_{Li} = \log d(t_{pi}), i = 1, 2, 3.$$

The detected peaks of 8 independent complete scan samples are marked in Fig. 8. The CEP time interval that corresponds to 240 ms in time domain is shifted to 0. Although some of the curves appear close to each other in the figure, their peak-indexed CEP time interval features actually span a considerable wide range in the feature space, as shown in Fig. 5 and the zoom-in sub-figure in Fig. 8. In the zoom-in figure, we see that the indexed time interval values have little intra-class variability and inter-class similarity. For example, although "C3" and "C6" are from the same brand, their peak indexes deviate from each other with a distance up to 20, while the three peak indexes of device "C6" are consistent during the three independent scans that are collected in a wide time span.

2) CEP value features: QID also fingerprints the controller of a PRx based on the statistic of the recorded CEP values during a complete scan. What the CEP value differs from the CEP time interval is that the former can only take integer values. We analyze CEP values using probability mass function (PMF). Specifically, we compute the PMF of the CEP values as

$$p(V) = \frac{\sum_{i=1}^{N} 1\{v_i == V\}}{N}$$

where V is the absolute CEP value, ranging from 0 to 127, $1\{a == b\}$ is an indicator function, and N is the total number of CEPs in a scan sample. In particular, N is chosen to be one of the CEP value features because it is an indirect measurement of the frequency that the PRx sends CEPs. As we note in Section V-A, it is a PRx controller implementation specific feature rather than the oscillator drift. We find that the PMF of the CEP values is not a good feature, as the PMFs span a wide range of [0, 127], which introduces high variance in the output features. We also observe that even the same device may generate different PMFs within different scan samples. To reduce the variance in the output CEP value features, we further group the CEP values into 7 ranges. For each range, the probabilities of the CEP values within the range are summed up. Specifically, the 7 ranges are: 0, [1, 5], [6, 10], [11, 20], [21, 30), 30, (30, 127]. We choose these ranges empirically by correlating the statistical patterns of the ranges with the device brand.

Table II summarizes the features used in classification. These features are collected in both steady charging states and operating point switching transient states, through which QID extracts the fingerprints in the oscillator, coil and the controller for classification. We envision that the PRx controller features can separate the device brand and the oscillator features can then further separate the devices within the same brand.

B. Classification

The QID server classifies Qi-compliant devices by an ensemble classifier, also known as "bagging classifier", comprising of Support Vector Machine (SVM) [37], AdaBoost [38] with decision tree as weak learner, decision tree classifier [39], k-Nearest Neighbor (kNN) [40], and Linear Discriminant Classifier (LDC) [41]. The bagging algorithm in our design utilizes a voting system. When 3 or more classifiers are outputting the same device label, the bagging classifier chooses it as the final decision. Otherwise, the output of the SVM is chosen because it has relatively higher accuracy than the other classifiers.

The QID server stores all the extracted features and their corresponding device labels in a feature table. Then it utilizes the repeated random sub-sampling cross-validation, also known as Monte Carlo cross-validation [42], to split the data into training and testing set randomly. Finally, the classifier models are trained with the training set and validated with the testing set. The mean and the standard deviation of accuracy from the results of the sub-sampling experiments are recorded. It is shown [43] that cross validation evaluation introduces neighborhood bias to the time-continuous sliding window frame data, which results in overly optimistic model evaluation estimations. However, in our experiments each of the samples are collected in a wide span in time domain. In other words, all the features extracted from a complete experiment scan are independent to each other. As a result, the cross validation is suitable for our experiments.

The objective of the QID server is focused on classifying the device into one of the known classes. This design is applicable to the scenarios where the devices are already fingerprinted. For instance, a company may register and fingerprint all the work devices of employees, and then use QID to track the location of each device. However, QID can be easily extended to recognize new devices via online learning. For instance, by setting a detection threshold in the classifier, QID can identify whether the newly collected sample corresponds to any device that is already recorded. If the sample's probability of corresponding to an existing device is low, QID can recognize the device as a new one.

The QID server stores all the extracted features and their corresponding device labels in a feature table. Then it utilizes the repeated random sub-sampling cross-validation, also known as Monte Carlo cross-validation [42], to split the data into training and testing set randomly. Finally, the classifier models are trained with the training set and validated with the testing set. The mean and the standard deviation of accuracy from the results of the sub-sampling experiments are recorded.

The objective of the QID server is focused on classifying the device into one of the known classes. This design is applicable to the scenarios where the devices are already fingerprinted. For instance, a company may register and fingerprint all the work devices of employees, and then use QID to track the location of each device. However, QID can be easily extended to recognize new devices via online learning. For instance, by setting a detection threshold in the classifier, QID can identify whether the newly collected sample corresponds to any device that is already recorded. If the sample's probability of corresponding to an existing device is low, QID can recognize the device as a new one.

VIII. Implementation

In this section, we present the implementation of the QID system. Fig. 9 shows a QID sensor prototype.

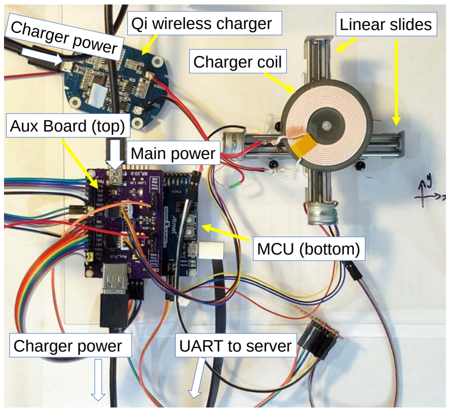


Fig. 9. A prototype of the QID sensor.

QID sensor base station. The base station is the mechanical component of the QID sensor, which is built on a clear acrylic board. The two stepper motor linear sliders enable the motion along the x and y direction respectively, with a moving distance of 90 mm each. A switch is added to one end of each screw, such that the MCU can reset the position each time the system is powered on. Another clear acrylic board (not shown in the figure) supported by four nylon hex spacers is the surface that the mobile device is put on. The cost of the mechanical components is less than \$20. Therefore it is feasible to be massively deployed in the public area. We note that the dimensions of the motion unit can be further reduced by adopting other machenical structure, such as transitional planar cable-driven movement system [44].

Embedded controller and motor driver. At the center of the QID sensor, the Atmel SAMG53N19 [45] MCU is employed, which is responsible for decoding and timestamping packets, driving the stepper motors, and sending collected data to the QID server. The MCU supports the UART communication with the server via a USB virtual COMM port. The motor driver IC is Toshiba TB6612FNG, which shares the power with the Qi PTx. The peak motor driving current is around 150 to 200 mA, which is negligible to the Qi wireless charging system because a typical COTS USB charger can provide 2000 mA current at 5 V.

Qi-compatible power transmitter. We choose a COTS GMYLE Mini Qi Charging Pad as the PTx, which is connected to the MCU via a data flow debug pin. The PTx coil is extended with a pair of wires, which provides extra flexibility, such that the coil moves without dragging the charger circuit board around.

The QID server. At the server side, the feature extraction and classification modules are implemented using approximately 1,100 lines of Python codes, including the `pySerial` UART library for the QID sensor communication handler and the machine learning library `scikit-learn` [46] for classification.

TABLE III
THE MEASUREMENT DELAY IN THE QID SYSTEM

Symbol	Definition	Mean (s)	Std (s)
T_1	Coil symmetric axis alignment time	8.050	0.927
T_2	Fingerprinting time	55.478	6.092
T_3	Feature extraction time	0.147	0.006
T_4	Classification time	0.001	N/A

IX. EVALUATION

In this section, we present the performance evaluation of QID based on 52 Qi-compliant devices. We first present the evaluation settings and then discuss feature analysis, measurement delay analysis, classification accuracy, feature backward search, and accuracy breakdown test.

A. Evaluation Settings

We evaluate 52 Qi-compliant devices in total, including 7 Google Nexus 4 (labeled as "N") and 45 attachable PRx modules from 6 different manufacturers, including DigiYes, Hugchg, and RAVPower. We note that the ICs in these modules are widely used in mainstream mobile devices. For example, the Texas bq51013B in the DigiYes modules is also adopted by Google Nexus 5. For each device, we conduct 10 complete independent scans to collect fingerprints. In total, there are 520 scan samples. To simulate the users' device placement behavior in the real world, we alter the phone placement manually. Specifically, for the first scan, the phone is aligned with the x axis of the motion plane. For the next seven scans, the device is rotated counter-clockwise for 45° each time. For the last two scans, the phone is placed on the pad with a random angle. To quantify the contribution of the motion platform, we repeat this whole process for once *without* moving the charger coil with respect to the PRx. We consider this as our *baseline* and will discuss it in Section IX-C.

In classification, the training and testing split ratio is 7: 3. In other words, 7 out of the 10 samples for each device are randomly chosen to train the QID classifier model, and the remaining scan samples are for testing. Such a process is repeated for 10 times. The average accuracy and the standard deviation of each classifier are reported. In addition, the hyperparameters in all the implemented classifiers are tuned by the grid search. As discussed in Section VII-B, we assume all the devices are already fingerprinted and recorded in the database.

B. Measurement Delay

The measurement delay is defined as the time delay from the moment when the power receiver is booted to the moment that the server produces a device label. Specifically, the measurement delay T_M is

$$T_M = T_1 + T_2 + T_3 + T_4$$

where T_1 is the coil symmetric axis alignment time, T_2 is the fingerprinting time, T_3 is the feature extraction time, and T_4 is

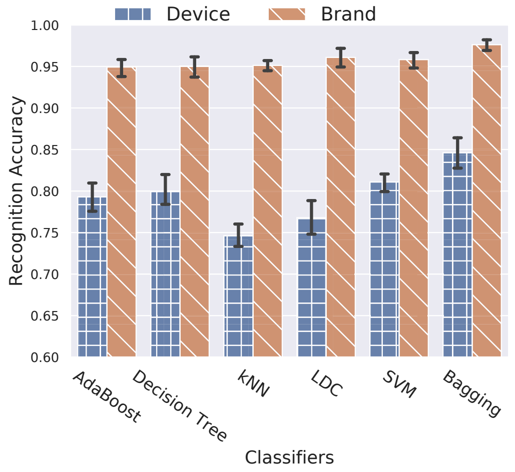


Fig. 10. The cross-validation score and device brand detection accuracy of different classifiers.

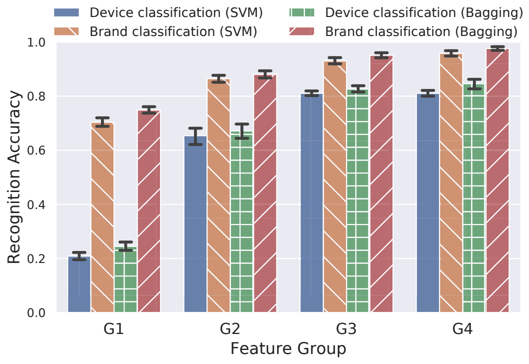


Fig. 11. The impact of feature selection on the classification accuracy. G1: classification without the CEP time interval features; G2: all features are included, but they are measured without the motion platform; G3: classification performance using the CEP time interval features only; G4: all features are included.

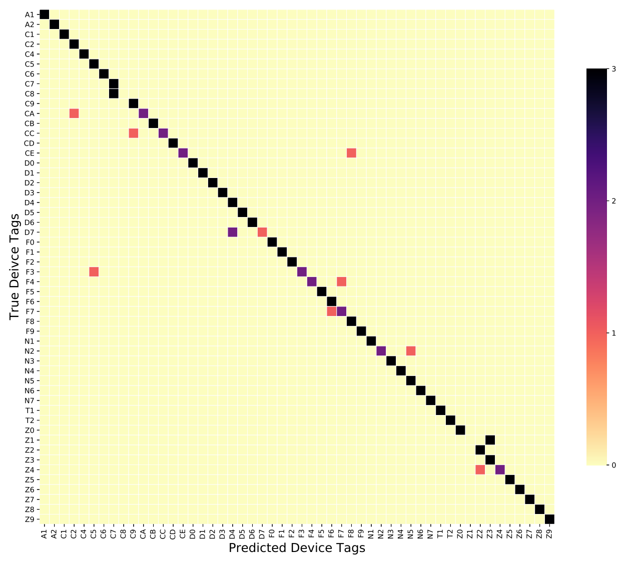


Fig. 12. Confusion matrix of the 52 evaluated devices.

the classification time. The means and standard deviations of these measurement delay terms are presented in TABLE III.

As one can see, the fingerprint phase time T_2 is around 55.5 s, which contributes the most to the total measurement delay. Thus reducing the time T_2 is crucial in further optimizing the measurement delay T_M.

The measurement delay is actually acceptable due to the characteristics of wireless charging. First, unlike other wireless communication systems where the user is usually in mobility, the charging process usually takes more than 10 minutes, during which the user device remains stationary. Second, in the targeted scenarios, such as a coffee shop that offers location and personalized services to customers, the users need to register their devices before using such user-identification service. During the registration process, QID can collect 8-10 different samples for future recognition. Finally, previous systems that exploit clock drifts for device identification have similar delay performance. For example, BlueID [21] takes 21 seconds for data traffic or 65 seconds for voice traffic to gurantee the low measurement error. in [19], it takes the system hours to collect enough packets in order to distinguish deivces. Therefore, the 60-second measurement delay in QID is actually acceptable.

C. Classification Accuracy

We first present the overall test accuracy of the cross-validation study. The overall accuracy is the ratio of the number of correctly classified scan samples to the size of the test set. Fig. 10 shows the means and standard deviations of the implemented classifiers from 10 repeated random sub-sampling cross-validation folds. As shown in the figure, all of the implemented classifiers can recognize the device brand with a mean validation accuracy up to 96.1%. Particularly, the bagging classifier identifies the brand with up to 97.9% mean accuracy. The mobile device brand classification accuracy is of interest because of the following two reasons. First, it is the foundation of device recognition. As mentioned in Section V-A, although the CEP interval is a good device separator, it confuses some devices from different brands. If the device brand is successfully identified, QID server can reduce the range of device candidates and increase the overall device classification accuracy. Moreover, brand recognition can enable applications like device brand specific advertisement. For device identification, the bagging classifier achieves an average accuracy of 85.2%. The highest accuracy achieved by QID is 89.7%. To illustrate the classifier performance, a confusion matrix is plotted in Fig. 12. Generally, the misclassified samples are from the devices of the same brand that have close clock drifts. For instance, there are two devices, namely “C8” and “Z3”, whose all 3 test samples are mis-classified. However, some devices are classifed into a different brand due to their close values in feature space.

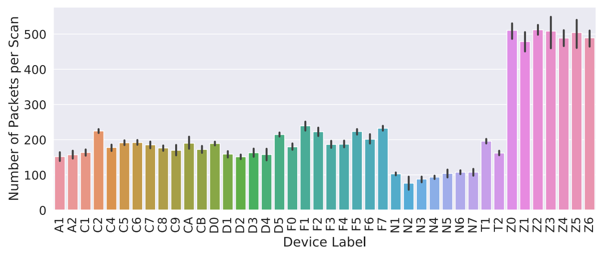


Fig. 13. Number of packet per scan feature distribution of 42 devices (10 samples per device).

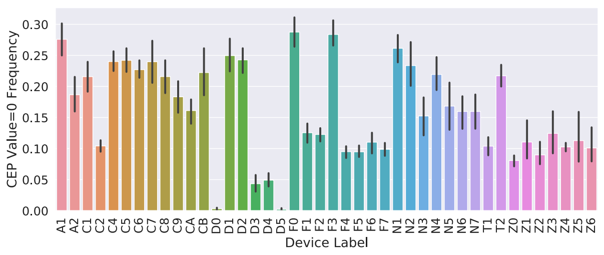


Fig. 14. The frequency of CEP value equaling 0 distribution of 42 devices (10 samples per device).

Next, we quantify the performance of the motion platform, namely the multi-coil array. The G2 in Fig. 11 shows the baseline result, where the motion unit is not enabled. Each device is sampled without the motion control for 55 seconds, corresponding to the delay T_2 in Section IX-B. We plot the classification accuracy of QID in Fig. 11 G4 for comparison. As we can see, the device recognition rate increases by 17% by both SVM and bagging classifier when the motion platform is enabled. The brand recognition accuracy is also boosted by 8%. It is indicated that the motion platform plays an important role in achieving reliable and sufficient device features for classification due to its ability to extract fingerprints from more spots, which captures a more complete profile of a device.

D. Impact of Feature Selection

Not all features are equally important. Fig. 13 and Fig. 14 shows the distribution of two features respectively, obtained from 42 out of the 52 devices, namely the total number of packets per scan and the frequency of the CEP value 0. The distribution of other CEP value frequencies yields similar trends as shown in Fig. 14 and is thus omitted. These two features contain more noise than the CEP interval (as shown in Fig. 5). Nonetheless, intuitively, these features are able to separate device classes to some extent. We next conduct the backward search to evaluate the effectiveness of each selected feature.

We perform two case studies. In the first case "G1", we evaluate the bagging classifier without the CEP time interval features, i.e., the onboard oscillator fingerprints. In other words, only the CEP value features are used. In the second case "G3", we evaluate the bagging classifier with only the CEP time interval features. The results of these two case studies are shown in Fig. 11. We observe that the device recognition accuracies of both the bagging and the SVM classifier degrade to about 21% in G1, which indicates the significant contribution of the onboard oscillator fingerprint to device identification performance. Another observation is that, although the CEP value features fail in device recognition, they are still able to distinguish the brands with 75% accuracy by the bagging classifier. Next, we compare G3 and G4. We can see that both the device and brand recognition accuracies of the bagging classifier are improved by about 2.5% by adding the CEP value features to the CEP time interval features. This indicates that although the CEP value features are not as important as the onboard oscillator fingerprints, it helps QID to reduce uncertainty and achieve higher accuracy. However, as the number of devices increases, the chance of CEP interval (PRx oscillator) feature overlapping is expected to increase. In such a case, the PRx controller fingerprints will provide necessary device brand identifies, thus reducing the collision in the feature space.

X. Discussion and Conclusion

In this paper, we present our design and implementation of QID, the first system that recognizes Qi power receiver during wireless charging using fingerprints from the onboard oscillator, coil characteristics, and control scheme of the wireless charging system. QID also employs a movement unit to emulate multi-coil power transmitter and allow for fine-grained fingerprinting. Our evaluation results show that QID achieves a high overall accuracy of both device and brand recognition. Therefore, we demonstrate the feasibility of leveraging public wireless charging infrastructure for tracking mobile users and providing ID/location-based services. Our results also open up new research questions on how to prevent the leakage of user's location with the increasing wireless charging station deployment in public.

QID has several limitations. First of all, unlike other wireless communication systems where passive remote sensing and recognition is possible, QID adopts a user-initiated device recognition approach. This narrows its applications because it requires the physical contact between the device and the sensor. However, this could also be an advantage because it preserves the user's awareness and thus protects the user's privacy. Second, at this stage QID requires motion parts to achieve fine-grained fingerprinting. We envision to achieve device recognition in wireless charging using only stationary multicoil array. However, since the commercially-available multicoil chargers are not ready for device recognition yet, our QID sensor implementation mainly focuses on emulating a multicoil array with motion control. We expect no motion unit is needed in the future implementation and deployment. Nevertheless, the mechanical structure could be possibly redesigned to achieve a smaller form factor. Finally, we note that the charging process may cause several short-time charging disruptions (about 1 to 2 seconds each) due to the discoupling between the PTx and PRx coils. In such a case, the user experience may be potentially impacted. However, as discussed above, the measurement delay is about 55 seconds. After a device is successfully identified, the PTx coil will move to a position (or switch to a particular physical coil) that achieves the maximum coil coupling to continue the power delivery process.

Our findings have important implications on the user privacy. The location of a mobile device may be tracked when the user charges mobile devices in public wireless chargers. Mitigating such possible user privacy breach is left for future work.

In the future, we plan to design a compact coil antenna to extract the data directly from the wireless power interface, such that the QID sensor can be non-intrusive to the PTx. We will also explore new fingerprinting trajectories to further reduce the measurement delay. Although the users usually initialize their mobile device registration by themselves in our targeted scenarios, we still aim to design efficient online machine learning algorithms to classify unknown devices, such that QID provides an easy-to-use interface and enablea a wider range applications. We can achieve this by quantifying the similarity between the incoming sample features with the ones already in our database. If the difference exceeds a threshold, QID determines the sample belongs to a device that has not been seen before.

ACKNOWLEDGMENT

This work is supported by the National Nature Science Foundation of China under Grant No. 61672282, and partially supported by the National Science Foundation under Grant Numbers CNS-1815274 and CNS-1943396.

REFERENCES

[1] W. P. Consortium, "Qi wireless charging goes mainstream," http://www.air-charge.com/news/21/19/Qi-wireless-charging-goes-mainstream, 2017, accessed: 2019-01-09.

[2] A. Charlton, "Wirelessly charge anywhere with these qi-enabled tables, lamps, speakers and accessories," https://www.gearbrain.com/qi-wireless-charging-tables-lamps-2528825500.html, accessed: 2018-07-25.

[3] D. Prindle, "Impress your guests (and top off their phones) with this diy wireless charging table," https://www.digitaltrends.com/how-to/diy-wireless-charging-table/, accessed: 2018-07-25.

[4] I. Markit, "Global shipments of wireless power receivers and transmitters to reach 2.1 billion in 2023, ihs markit says," https://tinyurl.com/qi-device-sales-2018, 2019, accessed: 2019-10-20.

[5] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The active badge location system," *ACM Transactions on Information Systems (TOIS)*, vol. 10, no. 1, pp. 91–102, 1992.

[6] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4. ACM, 2015, pp. 269–282.

[7] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of the 6th annual international conference on Mobile computing and networking*. ACM, 2000, pp. 32–43.

[8] A. W. T. Tsui, W.-C. Lin, W.-J. Chen, P. Huang, and H.-H. Chu, "Accuracy performance analysis between war driving and war walking in metropolitan wi-fi localization," *IEEE Transactions on Mobile Computing*, vol. 9, no. 11, pp. 1551–1562, 2010.

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

[10] H. Shin, Y. Chon, Y. Kim, and H. Cha, "Mri: Model-based radio interpolation for indoor war-walking," *IEEE Transactions on Mobile Computing*, vol. 14, no. 6, pp. 1231–1244, 2015.

[11] F. Ijaz, H. K. Yang, A. W. Ahmad, and C. Lee, "Indoor positioning: A review of indoor ultrasonic positioning systems," in *Advanced Communication Technology (ICACT), 2013 15th International Conference on*. IEEE, 2013, pp. 1146–1150.

[12] J. Xiong and K. Jamieson, "Arraytrack: a fine-grained indoor location system." Usenix, 2013.

[13] Y. Li, Y. Cheng, X. Li, Y. Wang, G. Xing, and X. Jiang, "Qiloc–a qi-wireless based platform for robust user-initiated indoor location services: Demo abstract," in *Proceedings of the 1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*, ser. BuildSys'14, 2014, pp. 184–185.

[14] *The Qi Wireless Power Transfer System Power Class 0 Specification*, 1st ed., Wireless Power Consortium, February 2017, parts 1 and 2: Interface Definitions. [Online]. Available: https://tinyurl.com/qi-spec-dl

[15] V. Brik, S. Banerjee, M. Gruteser, and S. Oh, "Wireless device identification with radiometric signatures," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*. ACM, 2008, pp. 116–127.

[16] O. Abari, D. Vasisht, D. Katabi, and A. Chandrakasan, "Caraoke: An e-toll transponder network for smart cities," in *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4. ACM, 2015, pp. 297–310.

[17] B. Danev and S. Capkun, "Transient-based identification of wireless sensor nodes," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*. IEEE Computer Society, 2009, pp. 25–36.

[18] R. Eletreby, D. Zhang, S. Kumar, and O. Yağan, "Empowering low-power wide area networks in urban settings," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 309–321.

[19] T. Kohno, A. Broido, and K. C. Claffy, "Remote physical device fingerprinting," *Dependable and Secure Computing, IEEE Transactions on*, vol. 2, no. 2, pp. 93–108, 2005.

[20] M. Cristea and B. Groza, "Fingerprinting smartphones remotely via icmp timestamps," *Communications Letters, IEEE*, vol. 17, no. 6, pp. 1081–1083, 2013.

[21] J. Huang, W. Albazrqaoe, and G. Xing, "Blueid: a practical system for bluetooth device identification," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 2849–2857.

[22] H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh, "Mobile device identification via sensor fingerprinting," *arXiv preprint arXiv:1408.1416*, 2014.

[23] A. Das, N. Borisov, and M. Caesar, "Do you hear what i hear? fingerprinting smart devices through embedded acoustic components," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014, pp. 441–452.

[24] J. Zhang, A. R. Beresford, and I. Sheret, "Sensorid: Sensor calibration fingerprinting for smartphones," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 638–655.

[25] A. Das, N. Borisov, and M. Caesar, "Tracking mobile web users through motion sensors: Attacks and defenses." in *NDSS*, 2016.

[26] X. Lu, D. Niyato, P. Wang, D. I. Kim, and Z. Han, "Wireless charger networking for mobile devices: Fundamentals, standards, and applications," *IEEE Wireless Communications*, vol. 22, no. 2, pp. 126–135, 2015.

[27] *The Qi Wireless Power Transfer System Power Class 0 Specification*, 1st ed., Wireless Power Consortium, February 2017, part 4: Reference Designs. [Online]. Available: https://www.wirelesspowerconsortium.com/knowledge-base/specifications/download-the-qi-specifications.html

[28] W. P. Consortium, "Magnetic resonance and magnetic induction - what is the best choice for my application?" https://tinyurl.com/wireless-charging-choices, 2017, accessed: 2019-02-13.

[29] *Freescale Wireless Charging ICs, MWCT1000CFM, MWCT1200CFM, MWCT1101CLH*, NXP Semiconductors, 5 2014, rEV 1.

[30] *Application brochure - Wireless charging for consumer*, Infineon Technologies, 3 2018, rev 4.0.

[31] *AN32258A: Intergrated Wireless Power Supply Receiver, Qi (Wireless Power Consortium) Compliant*, Panasonic Corporation, 10 2014, rev. 2.00.

[32] *MWPR1516: Higher integration receiver controller MCU for wireless power transfer application*, NXP Semiconductors, 1 2015, rev. 2.00.

[33] *BD57011AGWL: A stand-alone integrated IC for wireless power receiver*, ROHM Semiconductor, 1 2018, rev. 003.

[34] S. Bluetooth, "Bluetooth 4.2 core specification," *Bluetooth SIG*, 2009.

[35] "Ieee standard for low-rate wireless networks," *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pp. 1–709, April 2016.

[36] *bq51013B Highly Integrated Wireless Receiver Qi (WPC v1.2) Compliant Power Supply*, Texas Instruments, 3 2018, rEVISED MARCH 2018.

[37] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*. ACM, 1992, pp. 144–152.

[38] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*. Springer, 1995, pp. 23–37.

[39] L. Breiman, *Classification and regression trees*. Routledge, 2017.

[40] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[41] T. M. Mitchell, "Machine learning. 1997," *Burr Ridge, IL: McGraw Hill*, vol. 45, p. 995, 1997.

[42] W. Dubitzky, M. Granzow, and D. P. Berrar, *Fundamentals of data mining in genomics and proteomics*. Springer Science & Business Media, 2007.

[43] N. Y. Hammerla and T. Plötz, "Let's (not) stick together: pairwise similarity biases cross-validation in activity recognition," in *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*. ACM, 2015, pp. 1041–1051.

[44] T. Li, X. Tang, and L. Tang, "Algebraic expression and characteristics of static wrench-closure workspace boundary for planar cable-driven parallel robots," *Advances in Mechanical Engineering*, vol. 8, no. 3, p. 1687814016638217, 2016.

[45] *SAM G53N SMART ARM-based Flash MCU Datasheet*, 11240th ed., Atmel Corporation, July 2015, 32-bit ARM Cortex-M4 RISC processor. [Online]. Available: https://tinyurl.com/samg53-ds

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.