On the Feasibility of Using Two Mobile Phones and WLAN Signal to Detect Co-Location of Two Users for Epidemic Prediction

Khuong An Nguyen, Zhiyuan Luo and Chris Watkins

Abstract An epidemic may be controlled or predicted if we can monitor the history of physical human contacts. As most people have a smart phone, a contact between two persons can be regarded as a handshake between the two phones. Our task becomes how to detect the moment the two mobile phones are close. In this paper, we investigate the possibility of using the outdoor WLAN signals, provided by public Access Points, for off-line mobile phones collision detection. Our method does not require GPS coverage, or real-time monitoring. We designed an Android app running in the phone's background to periodically collect the outdoor WLAN signals. This data are then analysed to detect the potential contacts. We also discuss several approaches to handle the mobile phone diversity, and the WLAN scanning latency issue. Based on our measurement campaign in the real world, we conclude that it is feasible to detect the co-location of two phones with the WLAN signals only.

Keywords Epidemic tracking · Co-location · WLAN tracking

1 Introduction

In the past decade, mobile phones and the internet have become integrated into daily human life. More importantly, the wireless infrastructure has improved significantly in recent years to help transferring the information amongst the devices.

K.A. Nguyen (✉) · Z. Luo · C. Watkins
Department of Computer Science, Royal Holloway, University of London,
Surrey TW20 0EX, UK
e-mail: khuong@cantab.net; me@khuong.uk

Z. Luo
e-mail: zhiyuan@cs.rhul.ac.uk

C. Watkins
e-mail: chrisw@cs.rhul.ac.uk

G. Gartner and H. Huang (eds.), *Progress in Location-Based Services 2014*,
Lecture Notes in Geoinformation and Cartography,
DOI 10.1007/978-3-319-11879-6_5

People leave real-time digital footprints everywhere. These footprints may be used to track and study a disease in an epidemic. However, this information is largely unexplored in the public health research community so far. Since most people have a mobile phone, a contact between two persons can be regarded as a handshake between the two phones. Our task is to detect when the two mobile phones are close. In this paper, we propose a new methodology to allow the user to passively discover his contacts with other people in an epidemic. Our approach does not require a map or real-time GPS signals.

The key features of our approach are:

- We detect co-location—when two phones are close together without detecting exactly where the two phones are.
- This is done passively by an App on the user's smart phone. No signals are sent out.
- The users have full control of whether to track themselves.
- The tracking process uses existing Wi-Fi Access Points, and we have tested it in challenging, uncontrolled city environments.

The paper first explains how our idea can be applied into the epidemic tracking purpose. Then we discuss several wireless signal candidates for our system, and explain how to collect such signal on an Android phone. The properties and the challenges of using the signal are discussed. Finally, we conclude our findings and outline the future work.

2 Wireless Tracking for Epidemic Detection

2.1 Co-Location Tracking with Mobile Phones

Co-localisation is the process of identifying if two persons are in the same position at the same time. If they are co-located, there is a possibility that one can be infected by the other's disease. Given a time-stamp, we can keep track of the disease spreading history. Imagining a network of registered participants, where each patient uses his mobile phone to input his current symptoms. This information can be later uploaded onto a central server, and the system works out the probability of what disease he was infected. To discover the origin of an unknown disease, the doctor back-tracks the patients' contact history with other registered people in the same system.

Since most people have a mobile phone, a physical contact between two persons can be regarded as a handshake between the two phones. When such contact happens, the handsets must be close to each other. With our idea, we need not maintain a map for the devices, nor require knowing the exact location of the phone at any moment. The remaining question is: 'How can we detect when the mobile phones are close?' Given a particular time-stamp, we need a unique and ubiquitous property to reliably match any two mobile phones' location, and recording that a physical contact has happened. Fortunately, there are many wireless signals such as WLAN, Bluetooth, GSM, FM that are available in many places and can be freely captured. The next section discusses the pros and cons of these wireless signals for our project.

2.2 The Wireless Signals Candidates

Signal availability is the most important criterion to decide which wireless signal we will use. The signal should cover both indoor and outdoor spaces. Two standout candidates were the WLAN signal, and the GSM signal.

The GSM signal coverage has increased significantly in recent years, thanks to the wide deployment of many cellular phone towers (Ibrahim and Youssef 2013). Unfortunately, the Android NeighboringCellInfo class used to access GSM cellular information is phone-dependent and network-dependent. Many Samsung phones did not work in our experiments.

The WLAN signals are popular indoor, but were not so popular outdoor a few years back. Thanks to the increasing number of outdoor WLAN Access Points (APs), most notably the recent BT-Fon network, which allows the home router to transform into a public wireless hotspot. There have been over 5 million available APs in the UK since 2012, with 20,000 new hubs being added weekly (represented by the red and blue dots in Fig. 1). In our experiments, there are always at least 10 available APs at any position. Many areas in the city centre of London have more

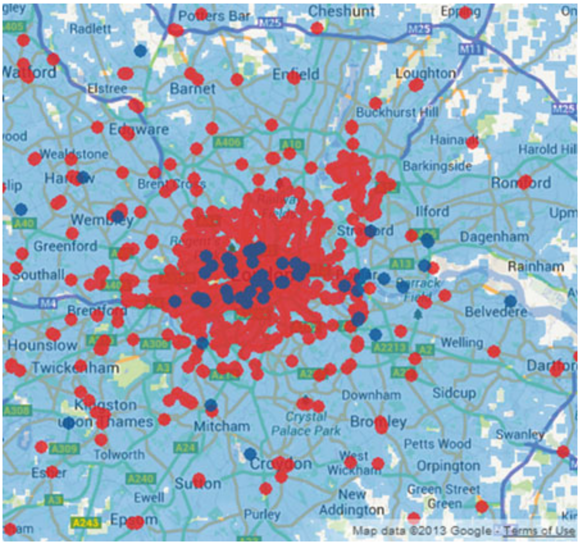


Fig. 1 BT WiFi hotspots in London

than 30 accessible APs. Another example is the commercialised Skyhook project,[1] which provides a world-wide WiFi RSSI signal strength to physical location map to alleviate the need of GPS coverage.

Some popular signals such as Bluetooth or infrared have restricted range, and are not popular outdoors. Other signals such as FM are available, but require additional decoder on the handset to read them. Such decoders are not widely supported by current smart phones. Thus, the WLAN signal is our best candidate. It is also possible to combine other signals with WLAN to increase the location's uniqueness (Pei et al. 2012).

2.3 An Android App to Collect the WLAN Signal Strength

To record the WLAN signals, we designed an Android app, which runs in the phone background to periodically scan the signals from the nearby APs every 30 s. This is the default scanning rate, which can be customised. We chose the Received Signal Strength Indicator (RSSI), which can be collected easily with the Android API.

The theoretical WLAN RSSI varies from 0 to −100 dBm, where higher value represents stronger signal. We looked into the Android source code, and found that RSSI equal or bigger than −55 dBm is considered the strongest signal, which is shown as a full bar of signal on the phone. If the measured RSSI is equal to or less than −100 dBm, the Android phone shows an empty bar. Based on this scale, we define three ranges of WLAN RSSI.

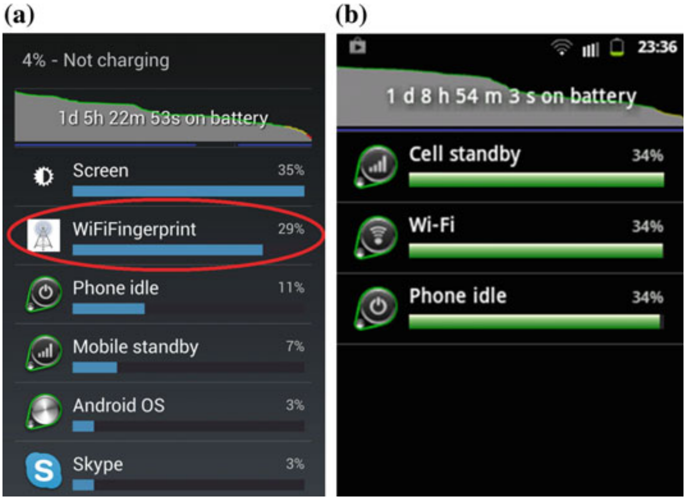
- RSSI from −55 to −70 dBm represent strong signals.
- RSSI from −70 to −85 dBm represent medium signals.
- RSSI from −85 to −100 dBm represent weak signals.

Under normal usage, our app consumed 29 % of the total power with our Google Nexus phone (Fig. 2a), and 34 % on our Galaxy Y phone (Fig. 2b).

2.4 How to Collect WLAN Signals on Android: Active Scanning or Passive Scanning?

There are two ways to collect the WLAN signals with an Android phone, active scanning and passive scanning. Both of them belong to the IEEE 802.11 MAC layer. In both cases, there is no authentication needed between the mobile device and the AP.

[1] http://www.skyhookwireless.com.



4% - Not charging

1d 5h 22m 53s on battery

Screen 35%

WiFiFingerprint 29%

Phone idle 11%

Mobile standby 7%

Android OS 3%

Skype 3%

23:36

1 d 8 h 54 m 3 s on battery

Cell standby 34%

Wi-Fi 34%

Phone idle 34%

Fig. 2 Battery consumption of WiFi scanning. **a** Galaxy Nexus. **b** Samsung Galaxy Y

With passive scanning, the phone constantly listens on consecutive channels for the beacons periodically sent by the APs. We looked into the Android source code, and found that the dwelling time on each channel is set at 120 ms. According to the IEEE 802.11 standard, the WLAN APs should send out beacons every 100 ms on all channels at the same time. Theoretically, the extra 20 ms interval should be sufficient for the mobile device to receive at least one beacon per channel. Thus, with 13 channels of the 2.4 GHz spectrum (in Europe), it takes at least 1,560 ms for an Android phone to scan all nearby APs theoretically.

In reality, the total scanning time also includes the information processing delay, in which the device processes the received beacons on each channel. Although passive scanning consumes less battery power, the device cannot pick up hidden APs, which are configured not to send out any beacon. Table 1 compares the passive scanning time in 5 h with our two phones.

With active scanning, the mobile device sends the probe request frames on all channels (similar to how the beacons are sent by the APs), and waits for the probe responses from the APs. The probe request frame can either contain a network name (SSID) of the AP the mobile device wishes to connect to, or an empty SSID, which all nearby APs should respond to. According to the IEEE 802.11 standard, the device should listen for a minimum of MinChannelTime (in ms) on a single

Table 1 Summary of passive scanning time

	Google Nexus (ms)	Samsung Galaxy Y (ms)
On average	5,022	5,016
Slowest single scan	5,194	5,641
Fastest single scan	4,909	4,902

Table 2 Summary of active scanning time

	Google Nexus (ms)	Samsung Galaxy Y (ms)
On average	962	1,419
Slowest single scan	1,045	1,532
Fastest single scan	944	1,400

channel. If no probe response is heard within this interval, the device assumes that this channel is empty, and moves on to the next one. If more than one probe response is heard within this interval, the device will continue to listen till the MaxChannelTime (in ms) has elapsed on the same channel. There is no strict definition of MinChannelTime and MaxChannelTime by IEEE 802.11 however. We looked into the Android open source code and found that Google implemented just a single dwelling time constant of 30 ms. For each probe request frame the device sends on the channel, there is an extra 3 ms delay. Table 2 compares the active scanning time in 5 h with our two phones.

In reality, laptops and other devices can decrease the scanning latency by forcing a scan on a specific channel only. Therefore, the device constantly listens on a particular channel, knowing that the AP should send out beacons on all channels. However, we cannot execute this method on non-modified Android firmware yet.

With either scanning setting, our app only wakes the CPU up every 30 s to scan the WLAN signals. This parameter is also customisable. While active scanning can complete faster and discover hidden APs, it consumes more power than passive scanning. A better option is to perform active scanning when the user is on the move, and switch to passive scanning when no movement is detected.

2.5 The Inverted System

Theoretically, it is possible to invert our system to have the APs track the mobile phones. This structure has the advantage of requiring no additional code on the phones, at the expense of a higher processing load on the APs, which have to track a huge amount of mobile users. In addition to the scalability issue, it is unlikely such changes can be done on the APs, without permission from the network providers.

On the security side, it would be undesirable for the users to be forcibly tracked by the APs. The users' anonymity is broken, because he can be identified by his GSM number or the phone's WLAN MAC address. In contrast to our original approach, the users can simply stop the tracking app on the phone without disrupting normal GSM or WiFi uses.

Table 3 Comparison of our system and FluPhone

	Our system	FluPhone
Technology	WLAN	GPS and Bluetooth
Contact detection	Off-line	Real-time
Battery consumption	Average	High
Always-on connection	No	Bluetooth
Custom code	Yes	Yes

2.6 *Related Work*

To the best of our knowledge, there was only one similar research known as the FluPhone project (Yoneki 2011). Both of our system and FluPhone aimed to provide mobile phone localisation for epidemic tracking. However, there are two key differences. First, we do not record the physical location of the users. Second, while FluPhone uses GPS and Bluetooth signal to discover nearby handsets in real-time, our approach analyses the off-line signal data to discover such contact. Table 3 compares the two approaches.

There were other work involved the WLAN signals for indoor and outdoor localisation (Wang et al. 2012; Chintalapudi et al. 2010; Martin et al. 2010). Yet, there was little attention to the co-localisation aspect, where the timing of the contact is important (Krumm and Hinckley 2004). The most notable use of WLAN localisation was fingerprinting, where real-time WLAN data are compared to a training database (Bahl and Padmanabhan 2000). Our work does not involve such training data, instead, the signal from different phones are compared directly based on its time-stamp.

3 Test Beds

We recorded the WLAN signals in two UK cities. Our first test bed was recorded near a busy railway station, where the second test bed was recorded on the streets of London. We used two Android mobile phones—the Google Nexus with the Jelly Bean firmware, and the Samsung Galaxy Y with the Gingerbread firmware. The ‘WiFi Fingerprinting’ app we designed to collect the WLAN signals can be downloaded on the Android app store.

4 WLAN Signal Properties

With our app, we collected the WLAN signal in different locations in the UK to assess the following criteria for both static and moving phones.

- Two co-located phones should observe similar WLAN signals.
- How the WLAN signals distinguish in different locations.

While other research investigated the indoor WLAN properties for fixed clients, our experiments looked at the signal properties for outdoor moving device, where the environment is not as stable as indoor. Further, we are more interested in how distinguishable multiple signal traits are, rather than from a single device perspective as in other work. Finally, we focus on the number of found APs, beside the signal strength. We used the RSSI as a quality measurement for the WLAN signal in our experiments.

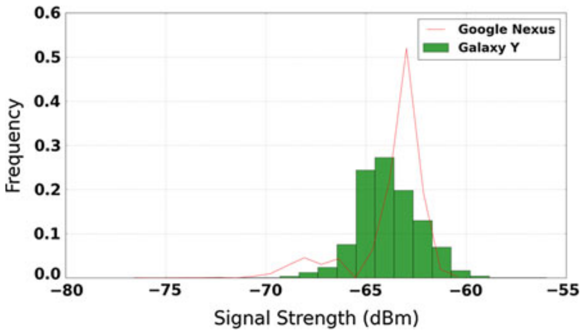
4.1 The WLAN Signal of Static Phones

When the two phones are co-located, we assume that they should hear the same signals from nearby APs. Figure 3 depicts the histogram distribution of the WLAN signals between our phones, which are positioned right next to each other, and an outdoor BT AP. We collected 6,130 signal readings in half a day. In our experiment, the signal variation was around 20 dBm, in contrary to the 10 dBm interval reported for indoor WLAN (Kaemarungsi and Krishnamurthy 2004). The maximum and minimum readings observed from Google Nexus phone were −60 and −77 dBm respectively, while Galaxy Y recorded −56 and −75 dBm. However, the majority of signals peak around the highest frequency RSSI in both phones. The most frequent RSSI was −64 dBm for Google Nexus, and −63 dBm for Samsung Galaxy Y.

Despite the total 20 dBm signal variation, the maximum signal difference at any moment between the two phones was under 11 dBm (Fig. 4). The signal difference was small at night, and became bigger by lunch time.

Figure 5 shows that the signal difference of the two phones was less than 4 dBm for 91 % of the time. They had the exact signal reading for 16 % of the time. The majority of the signal difference was just 1 dBm, 34 % of the time.

Fig. 3 Outdoor WLAN signals distribution



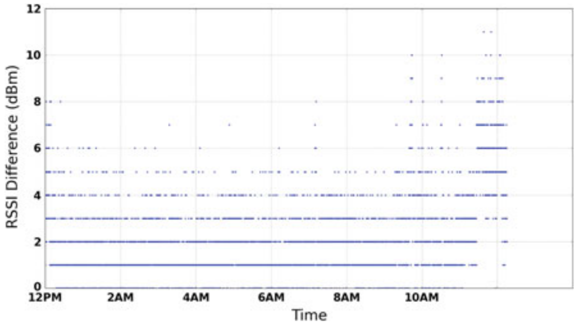


Fig. 4 Individual signal strength reading between two phones

4.2 The WLAN Signal of Moving Phones

As the user moves around, the scanning latency affects the similarity of the receiving signal in different locations. The latency is caused by the delay from the handset in sending the probe request frames, and the delay from the APs to reply with response frames. For example, AP_1 responses within the first 30 s, however, AP_2 and AP_3 response a second later, when the user has already moved to a new position.

In our experiment, two persons walked side by side with a mobile phone in the pocket. Both phones were synchronised to invoke 3 continuous scans every 30 s. Our Google Nexus phone took less than 1 s on average per scan, while the Samsung Galaxy Y phone took more than 1.4 s. The continuous scans help discovering the missing APs from previous scans. Figure 6 depicts the WLAN signal strength from the phones to the nearest fixed AP. Since we did not know the exact distance from the AP to our handset, we assumed that the starting point is the location where the strongest signal can be obtained, indicated by the zero point on the x-axis.

We observed that the variation is large for stronger signals, and decreases as the signals fade away. The signal was completely lost among all the ambient noises at a distance of 170 m, from the initial position. This result shows a much greater range with outdoor AP, compared to the relatively short 15–20 m distance for an indoor AP.

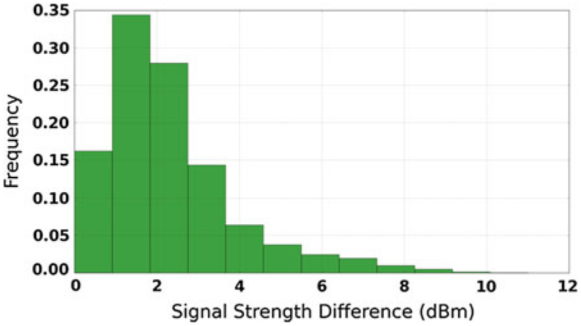
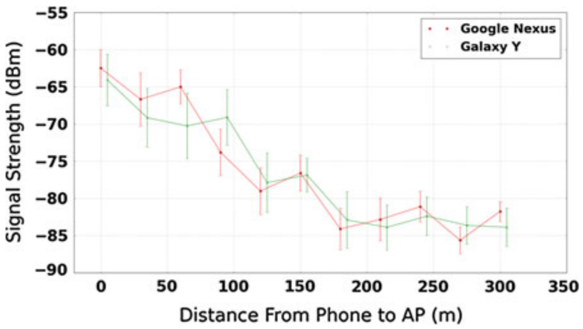


Fig. 5 Percentage of signal reading difference between two phones

Fig. 6 Correlation between WLAN signal strength and distance



We expect a rough 5–10 dBm difference between two consecutive locations 30 m apart, assuming they are in a relative straight line from the AP.

Since both phones were side by side in our experiment, we expect them to discover the same number of APs. However, *none of the 253 pairs of signal vectors between the two phones has the same number of detected APs*. This number highly contrasted with 3,107 pairs of WLAN signal vector with the exact number of detected APs, over 3,894 total pairs, recorded when the two phones are not moving in the previous experiment. This result confirms the scanning latency we suspected above.

4.3 Number of Detected APs

One of the indicators to distinguish the phone's location is the amount of detected APs. In our experiment, there were 412 APs recorded by the Google Nexus phone, and 554 APs recorded by the Samsung Galaxy Y phone. Of 252 APs found by both phones during the 20 min journey, the highest number of appearance of a single AP was 15 for Google Nexus and 13 for Samsung Galaxy Y (Table 4).

Figure 7 demonstrates the appearance of the top 10 commonly observed APs in the whole journey of both phones. Those APs appeared frequently in both phones' list of detected APs, when they are in the same position (started at 14:56). The Google Nexus phone has a stronger and newer antenna, therefore, it can discover more APs than the old Samsung Galaxy Y phone.

Table 4 Number of detected APs

	Google Nexus	Samsung Galaxy Y
Total number of scans	120	120
Total recorded APs	504	558
Commonly observed APs	252	252
Highest number of single AP found	15	13

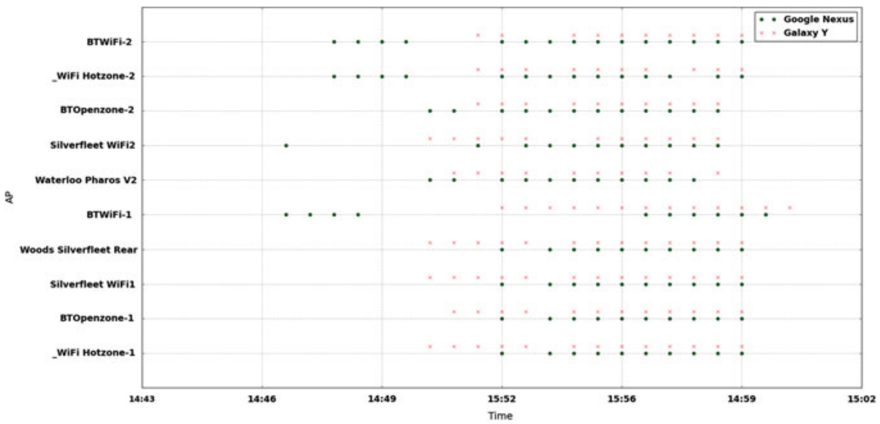


Fig. 7 Top 10 AP with highest frequency of appearance

5 Mobile Phone Co-Localisation with WLAN Signals

5.1 A Matching Rate Algorithm for Co-Located Phones

In our experiment, we recorded 1,643 APs in 1 h journey in London. We observed at least 10 APs at any given place, and more than 30 APs in the city centre of London. Based on such high number of APs, we define an algorithm to calculate a matching rate given any two RSSI vectors.

The two WLAN vectors are considered 100 % matched, if they have the same number of recorded APs, and for every AP recorded by Phone$_1$, it is also observed by Phone$_2$, and vice versa. If the two signal vectors share some APs in common, but also contain their detected APs. The matching rate is calculated as the number of common APs, divided by the total number of APs. A high matching rate means the two devices are close, and a low one means they are further away. Ideally, we aim to deliver a matching rate as close as possible to the physical distance.

The above algorithm worked relatively well for a large number of APs. When the number of nearby APs is low, we try to incorporate the WLAN signal strength into our equation. Our assumption is that when two phones are close, they should observe a strong signal from the same AP. In other words, if one phone sees a strong AP, while the other does not, they are unlikely to be close. However, this assumption is subjective, and although it works well in our experiment, certain location with different combinations of APs may not see a better result. We add up the signal strength of the common APs from the WLAN vector of both phones. The result is divided by the total signal strengths from all nearby APs of both devices.

In our experiment, two persons started at the same location, they then walked in different paths at 14:42, and re-joined midway at 14:56, and continued the journey until 15:01 (Fig. 8).

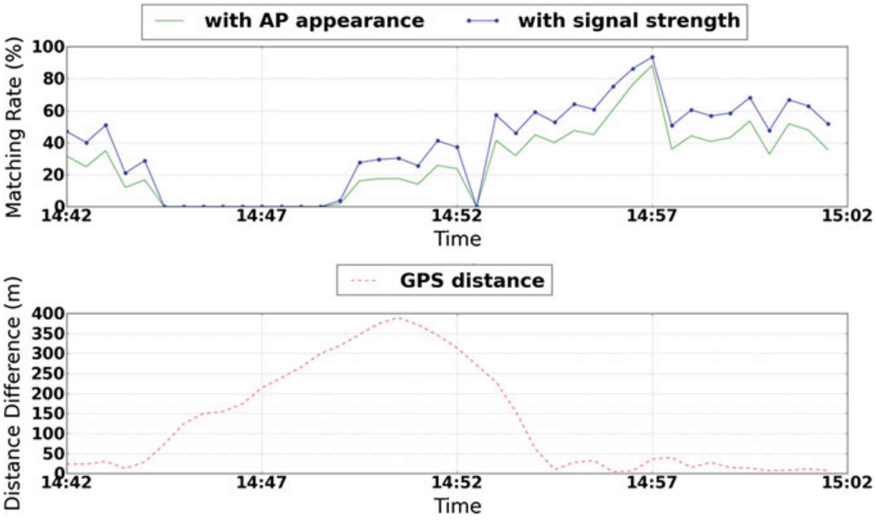
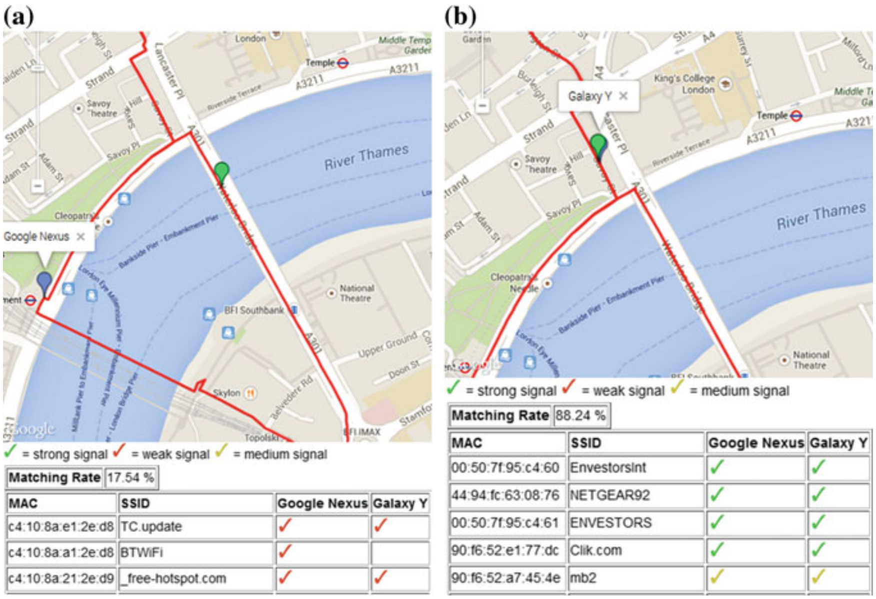


Fig. 8 Comparison of matching rate and GPS distance



(a)

✓ = strong signal ✓ = weak signal ✓ = medium signal

Matching Rate	17.54 %

MAC	SSID	Google Nexus	Galaxy Y
c4:10:8a:e1:2e:d8	TC.update	✓	✓
c4:10:8a:a1:2e:d8	BTWiFi	✓	
c4:10:8a:21:2e:d9	_free-hotspot.com	✓	✓

(b)

✓ = strong signal ✓ = weak signal ✓ = medium signal

Matching Rate	88.24 %

MAC	SSID	Google Nexus	Galaxy Y
00:50:7f:95:c4:60	EnvestorsInt	✓	✓
44:94:fc:63:08:76	NETGEAR92	✓	✓
00:50:7f:95:c4:61	ENVESTORS	✓	✓
90:f6:52:e1:77:dc	Clik.com	✓	✓
90:f6:52:a7:45:4e	mb2	✓	✓

Fig. 9 Mobile phones co-localisation on google maps. **a** Low matching rate for separated phones. **b** High matching rate for co-located phones

The matching rate started off positively at the beginning, when both phones were together. As the phones went in different routes, the matching rate dropped. Since both phones were in open space, they still saw some similar APs, despite their distance (Fig. 9a). However, the observed RSSI was weak. The matching rate

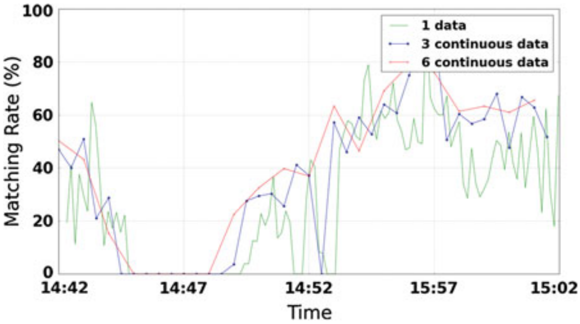


Fig. 10 Improvements from multiple continuous scans

increased as the phones approached each other, and remained stable for the remaining journey (Fig. 9b). When the phones are close, they both observe many strong RSSI. A real-time demo of our experiment on Google Maps can be viewed on our website.[2]

5.2 *Handling the Mobile Phone Signal Diversity*

Each manufacturer can implement their WLAN antenna, or WLAN adapter, which can potentially affect the receiving signal on the mobile device. A new mobile phone with a bigger and stronger antenna is able to discover long-distance APs, and receives stronger signals from nearby APs. There were multiple attempts to tackle the heterogeneous devices issue, which we classify into two broad categories, calibration-based and algorithm-based. In the first type, the new device is calibrated either manually or automatically within the system. Lee and Han (2012) use different known landmarks in the building to calibrate the devices' signal. With algorithm-based approach, Park et al. (2011) use a linear transformation model and kernel estimation were used to solve the problem, while Kjærgaard (2011) and Ibrahim and Youssef (2013) compare two pairwise signal vectors directly. In this paper, we explain our simple approach to normalise the WLAN signal data, since our goal is off-line co-localisation detection and we already have access to the full database through-out the phones' journey.

Our approach assumes that all registered devices may observe the strongest possible signal within their antenna capability. This is a fair assumption, given the vast number of APs, and the amount of the time the phone is observed. We normalise the signal strength readings for each mobile phone into a number between (0, 1), so that they are directly comparable to the signal strengths from other devices. Without loss of generality, given an RSSI vector representing the WLAN signal strength of N nearby APs, $RSSI = (s_1, s_2, \ldots, s_N)$, with s_i is the signal strength observed from AP$_i$, we divide each s_i by the strongest signal $s_{\max}$ observed in the whole journey.

[2] http://khuong.vn/Map.

$$\begin{aligned} RSSI_{norm} &= (\frac{s_1}{s_{\max}}, \frac{s_2}{s_{\max}}, \ldots, \frac{s_N}{s_{\max}}) \\ &= (s'_1, s'_2, \ldots, s'_N) \end{aligned}$$

5.3 Handling the Scanning Latency of Moving Phones

The movement and speed of the user have a strong impact on the AP discovery and the signal strength. We have shown that even when the two moving phones were side by side, none of our recorded signal traits was 100 % matched at any moment, based on the appearance of the APs. However, 79.8 % of the pairs had 100 % matching rate, when the phones were not moving. A simple approach is to increase the time window, and the continuous scan frequency to capture the missing APs. Since the user cannot move a long distance in a short period of time, and the WLAN signal strength was similar within 30 m in our experiment, we can combine multiple scans within 10–30 s, depending on the walking speed. We outline our scheme to combine *N* continuous WLAN scans.

- First, we generate a list of all APs found within N continuous scans.
- Second, we average the signal of the same APs found within these scans.
- Finally, we remove the APs with very weak signal (e.g. ≤−90 dBm).

Since a single active WLAN scan takes 900–1,500 ms in our experiment, the parameter *N* should not be too big to conserve battery power, but is also not too small to capture the missing APs. We decided to leave out the weak APs, because some old phones may not be able to see them, as with our Samsung Galaxy Y.

With our method, the number of perfect matching pair of WLAN vectors increased from 0 to 5. The number of 50 % matching pair of vectors increased from 46 to 114. We increased the majority of the matching rate to 40 % with our approach. Figure 10 shows a higher matching rate after combining 3 and 6 continuous scans.

5.4 Bringing It All Together

Figure 11 demonstrates the progress of our scheme for co-localisation detection, given the WLAN signal database from two phones. First, we normalise the WLAN signal strength with our device diversity handling scheme (Sect. 5.2). Several continuous scans are then combined to tackle the scanning latency (Sect. 5.3). In the processing phase, a matching rate is calculated, for each pair of WLAN vectors with the same time-stamp (Sect. 5.1).

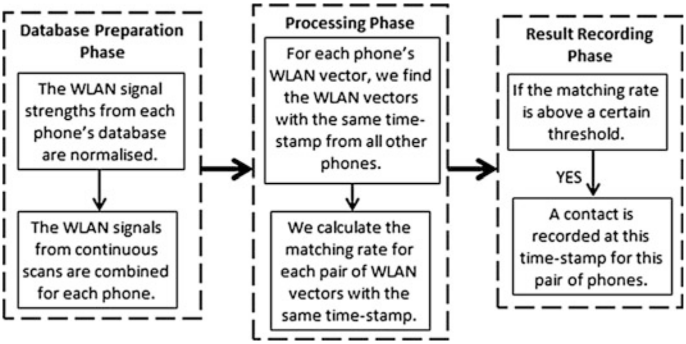

Database Preparation Phase
The WLAN signal strengths from each phone's database are normalised.
The WLAN signals from continuous scans are combined for each phone.
Processing Phase
For each phone's WLAN vector, we find the WLAN vectors with the same time-stamp from all other phones.
We calculate the matching rate for each pair of WLAN vectors with the same time-stamp.
Result Recording Phase
If the matching rate is above a certain threshold.
YES
A contact is recorded at this time-stamp for this pair of phones.

Fig. 11 The progress of our co-localisation scheme

6 Conclusion

We have demonstrated the feasibility of co-localisation detection using mobile phones and the public outdoor WLAN APs for the epidemic tracking purpose. We designed an Android app to collect the WLAN signals, and investigated their properties for co-localisation tracking. To evaluate our approach, we defined a method to compute matching rate, and compared it to the actual GPS distance. We tested our approach in a real, crowded environment to confirm that the matching rate closely reflects the GPS distance. We discussed our approach in handling the mobile device diversity by normalising the WLAN signals. We also identified the scanning latency which is the main cause of degrading the matching rate of moving phones. We showed that the matching rate can be improved up to 30 % by combining multiple continuous scans within a small time-window. Our future work is to investigate new ways of computing the matching rate even with a small number of Aps, and conduct a large scale experiment with many phones.

Acknowledgments The authors would like to thank the anonymous reviewers for their insightful comments on the paper. This research is funded by the Computer Science Department of Royal Holloway, University of London, and EPSRC grant EP/K033344/1 ("Mining the Network Behaviour of Bots").

References

Bahl P, Padmanabhan VN (2000) RADAR: an in-building RF-based user location and tracking system. In: Proceedings of nineteenth annual joint conference of the IEEE computer and communications societies, INFOCOM 2000, vol 2. IEEE, pp 775–784

Chintalapudi K, Padmanabha Iyer A, Padmanabhan VN (2010) Indoor localization without the pain. In: Proceedings of the sixteenth annual international conference on mobile computing and networking ACM, pp 173–184

Ibrahim M, Youssef M (2013). Enabling wide deployment of GSM localization over heterogeneous phones. In: IEEE international conference on communications, pp 6396–6400

Kaemarungsi K, Krishnamurthy P (2004) Properties of indoor received signal strength for wlan location fingerprinting, in mobile and ubiquitous systems: networking and services. The first annual international conference on obiquitous. IEEE, pp 14–23

Kjærgaard MB (2011) Indoor location fingerprinting with heterogeneous clients. Pervasive Mob Comput 7(1):31–43

Krumm J, Hinckley K (2004). The nearme wireless proximity server. In UbiComp: ubiquitous computing. Springer, Heidelberg pp 283–300

Lee M, Han D (2012) QRLoc: user-involved calibration using quick response codes for Wi-Fi based indoor localization. In: 7th international conference on computing and convergence technology (ICCCT). IEEE, pp 1460–1465

Martin E, Vinyals O, Friedland G, Bajcsy R (2010) Precise indoor localization using smart phones. In: Proceedings of the international conference on multimedia, pp 787–790

Park JG, Curtis D, Teller S, Ledlie J (2011) Implications of device diversity for organic localization. In: Proceedings of INFOCOM. IEEE, pp 3182–3190

Pei L, Liu J, Guinness R, Chen Y, Kroger T, Chen R, Chen L (2012). The evaluation of WiFi positioning in a bluetooth and WiFi coexistence environment. In: ubiquitous positioning, indoor navigation, and location based service (UPINLBS). IEEE, pp 1–6

Wang H, Sen S, Elgohary A, Farid M, Youssef M, Choudhury R (2012) No need to war-drive: unsupervised indoor localization. In: Proceedings of the 10th international conference on mobile systems, applications, and services, ACM, pp 197–210

Yoneki E (2011) Fluphone study: virtual disease spread using haggle In: Proceedings of the 6th ACM workshop on challenged networks, ACM, pp 65–66