# WiFi tracking of pedestrian behavior

Andreea-Cristina Petre , Cristian Chilipirea, Mitra Baratchi, Ciprian Dobre, Maarten van Steen

*University Politehnica of Bucharest, E-mail:* {*andreea.petre; cristian.chilipirea; ciprian.dobre*}*@cs.pub.ro*

*University of Twente, E-mail:* {*m.baratchi; m.r.vansteen*}*@utwente.nl*

**Summary**

Tracking pedestrian behavior is receiving increasingly more attention. Various techniques have been used so far, yet tracking through WiFi seems to be the most popular one. This popularity comes from the ubiquity of modern smartphones, of which it is known that most have their WiFi enabled all the time. In this chapter we concentrate exclusively on how this WiFi tracking works, and explain its potentials and pitfalls. Special attention is given to the quality of data from WiFi scanning devices, and how this data can, and should be cleaned up before attempts at extracting information from sets of detected devices. As an illustration of the power of WiFi tracking, we also briefly discuss a few recent results from gathering WiFi data from a large event that attracted over 100,00 people spread across three days.

*Keywords: WiFi; pedestrian; tracking*

## 1 Introduction

Pedestrian dynamics continues to receive much attention from scientists, architects, event organizers, game designers, and many other groups. The need for understanding those dynamics is undisputed in the face of questions related to safety, experience enhancement, and planning in general. To this end, many models have been developed, not only to capture observed behavior, but also to predict what will happen in given situations. In a recent survey, Wijermans et al. (2016) observe that by far most of these models lack proper validation for the simple reason that actual data sets were not readily available.

This lack of data on pedestrian behavior has changed dramatically in recent years. Notably with the massive introduction of smartphones, sensing what is happening in a crowd has become within reach. Not only has it become easier from a technological point of view, at least as important is that with no, or only minimal intrusion, has it become possible to sense behavior at large scales. In many cases, being able to actually measure behavior of *large* groups of people is a prerequisite for model validation.

Yet, it is not only the need for validating models that drives research into sensing pedestrian behavior. The fact alone that it can be done relative easily has triggered curiousity to further understand that behavior. In other words, even without interest in models, event organizers, urban planners, and so many others, simply want to know "what is going on," in order to take better informed decisions.

In this chapter we explore how WiFi technology can be deployed for measuring pedestrian behavior. The basic idea is simple: WiFi-enabled devices regularly send messages containing a unique identifier for that device. A WiFi scanner receives those messages, which can then be processed further. Having a unique

[*]Corresponding author or acknowledgement to supporting funding agency.

identifier enables us to (1) distinguish devices from each other, and (2) determine whether two scanners have observed the same device. In other words, we are capable of *detecting devices*. These two properties form the foundations for counting devices as well as tracking their locations when scanners are located at different places. If we know the ratio between devices and people, we can then draw conclusions on the movement of people.

Although other technologies exist as well, WiFi-based systems have become popular for two simple reasons. First, many people carry WiFi-enabled devices making it an ideal instrument for tracking people. Second, and very important, is that the owner of a device need not do anything to allow tracking except enabling the WiFi capabilities of her smartphone. In practice, smartphones virtually always have their WiFi enabled. Of course, this unintrusiveness with regard to tracking people imposes serious privacy issues for which often no easy solution is available.

Despite that the basic idea of WiFi-based tracking can be easily explained, accurately measuring pedestrian behavior this way is a nontrivial exercise. To make this clear, we start with presenting the technical principles of a WiFi-tracking system and pinpoint the sources of potential errors and how easily different failures in detections can be masked or corrected. Note that many researchers and practitioners tend to ignore the technology bottlenecks one has to solve, only to be surprised by the low quality of the resulting data set of detected devices. We illustrate some of the difficulties and inaccuracies of a WiFi-tracking process with our field findings obtained in a real-world tracking experiment.

No matter how good the equipment, WiFi-based detections have inherent quality problems caused by difficulties of using a radio-based medium. The resulting data set therefore always needs to be cleaned up. We present a set of filters that remove or correct detections leading to a higher quality data set that retains the relevant properties related to pedestrian behavior of the original set. In turn, this cleaning leads to better and faster analyses. Some of these filters are particular to WiFi but some of them can be applied to any other type of detection system.

Once we have a reasonable set of detected devices, there are many options. We illustrate some of these options in Section 4 by considering two different data sets. One comes from monitoring a three-day festival using some 25 scanners, which we use to discover likely paths taken by pedestrians. Another is based on scanning several locations for a long time with the purpose to *fingerprint* each location.

## 2 Principles of WiFi tracking

Tracking WiFi-enabled devices is based on the procedure which is meant for network discovery by mobile devices Curran et al. (2011) by a wifi access point(which we refer to as wifi scanner in the rest of this paper). There are two ways in which a connection can be established. First, a wifi scanner can advertise its presence by broadcasting beacon messages, to which a device can subsequently respond. However, instead of waiting for a scanner to announce itself, for mobile devices such as smartphones it is generally more efficient to actively seek for scanners. To this end, a mobile device periodically broadcasts a **probe request**. Such requests are sent regardless if a connection has been established: if a better scanner is detected, the mobile device will want to connect to that access point instead of its current one.

A probe request contains a lot of information, but most importantly, it contains the MAC address of the sending device and potentially a list of network identifiers known as SSIDs, through which the device has previously connected. Usually, the perceived signal strength of the probe request is available for the receiving device (i.e., scanner). A MAC address is uniquely associated with a network interface, and can

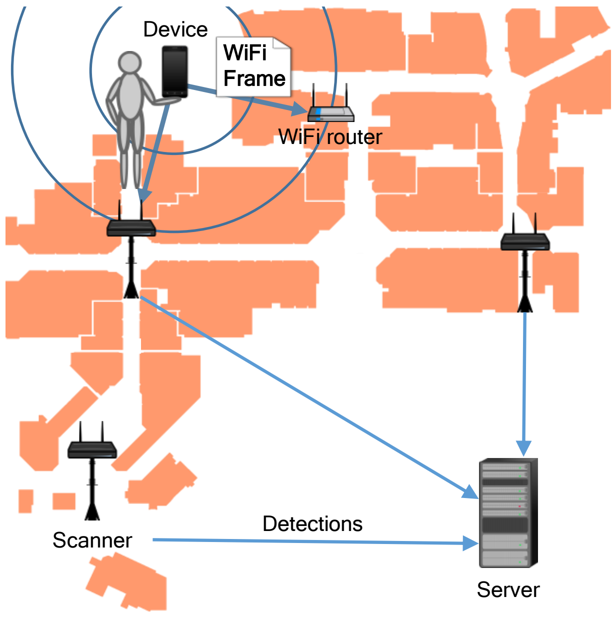


**Figure .1:** WiFi tracking

thus act as an identifier for a device. If a device has multiple network interfaces, it will thus, in principle, have multiple associated identifiers. We will denote the MAC address associated with the WiFi network interface of a mobile device as MID.

WiFi tracking works as follows. A scanner device with the identifier SID will receive a probe request from a mobile device with the identifier MID at time T. This can yield into triplets $\langle \mathsf{SID}, \mathsf{MID}, \mathsf{T} \rangle$. By chronologically ordering these triplets for a specific mobile device, we obtain a series of scanners that have subsequently detected that device. Knowing the location of the scanners should therefore give us information on a rough estimation about the whereabouts of the device, over a certain time span. As we explain in this chapter, matters are not that simple in practice.

## 2.1 WiFi scanners: technical background

WiFi tracking is based on the fact that smartphones and tablets are now ubiquitous. These **devices** have a WiFi module and are commonly used to access the internet. According to the Internet Society iso (2015) mobile broadband already accounts for more than 50% of traffic in developing countries. WiFi devices send frames which can be received and recorded by **scanners** (more details in Section 3.1). The scanners can be as simple as a household WiFi router with modified software that enables the recording of any WiFi frames the router antenna receives. The same thing can be achieved with a laptop, however WiFi routers have the lowest price. Using a WiFi frame the scanner can build what we call a **detection**, a $< scannerid; deviceid; timestamp >$ triplet. These triplets are sent to a central **server**. The process is represented in Figure .1.

A WiFi scanner is a device capable of receiving WiFi frames. Virtually any device with a WiFi interface (e.g., wireless routers or smartphones) can operate as WiFi scanner, provided it supports the so-called **monitor mode** as defined in the IEEE 802.11 standard iee (2012). In monitor mode, a device can capture any (correct) WiFi frames from any other device without the need for the two to have been first explicitly associated (and able to exchange further traffic) with each other. In practice, WiFi scanning is done by WiFi routers or dedicated devices (scanners).

A practical issue is handling the captured frames is transferring them to a processing server. One way or the other, probe request are meant for allowing exchange of traffic to a network through the scanner. To this end, a scanner is generally connected to a backbone network through a separate network interface. There are many variations and combinations used in practice. For indoor scanners it is often convenient to use a separate WiFi network, or an available (wired) Ethernet network. Outdoor scanners are often equipped with additional 3G and 4G interfaces which allow sending the data over mobile networks. When no network is available to transfer the data, it may also suffice to store captured frames locally on the scanner and to postpone data analysis.

## 2.2 Common issues

A perfect mobility data set would be one in which the location of a mobile device is accurately known at defined intervals. This means that there is no interval when the location of the mobile device is not registered. This does not mean that a device should not trigger detections at two or more scanners simultaneously, given an intersection in their coverage range. Simultaneous detections are acceptable as long as the RSSI values can be used to calculate a realistic positioning of the device.

Unfortunately, gathering a perfect data sets through tracking WiFi-enabled devices is a challenging problem. There are many sources of errors. Some the challenges are:

**Faulty scanner** Some errors are caused by the scanners and these are often the simplest to detect and correct. For example, any interval in which a scanner is shut down or cannot receive frames will generate a clear irregularity in the density of detections over time for that scanner. In our example data set, scanners automatically reboot once every 24 hours, leading to a noticeable glitch in the detections.

**Limitations of radio-based detections** WiFi uses a data transmission medium which is inherently unreliable Salyers et al. (2008). For example, most WiFi devices claim a 100-meter transmission range in ideal conditions. In reality, such specifications cannot be relied upon due to practical sources of impairment in wireless transmission, such as attenuation distortion, free space loss, noise, atmospheric absorption, multipath and refraction Beard and Stallings (2016). As an effect of such impairments, in practice it is observed that tunnels extend the transimssion range, while buildings and people are known to hinder transmissions. This also means that the shape or size of the area where WiFi frames can be correctly received can be highly irregular. Due to such issues, during our experiments we have come across the detection of a single device by five or more scanners at the same timestamp. However, such detections could not be explained considering the placement of scanners and a uniform detection range.

**Limitations of RSSI** Using trilateration based on the received signal strength indicator (RSSI), we should, in principle, be able to pinpoint the location of a device Liu et al. (2007). There are multiple problems to be addressed. First, RSSI measurements as taken by the scanners, are not standardized and can differ in value or strength across different types of scanners. Second, the signal strength itself can dramatically differ across multiple device manufacturers and even different devices of the same model. Solutions for the RSSI problems have been proposed Kim et al. (2012), but only for when the mobile device is the one taking the measurements. These solutions do not directly apply to the reverse scenario. An experimental evaluation of RSSI-based localization methods is presented in Zanca et al. (2008), illustrating its inherent difficulties.

**Timing errors** Scanners timestamp detections. Consequently, their clocks may introduce many inaccurate detections if not properly synchronized between different scanners. If the clocks are not synchronized one device moving from scanner A towards scanner B, may unexpectedly be associated with records showing that the detection at scanner B *followed* by a detection at scanner A. Even when the scanners are completely synchronized it may be difficult to determine the exact time a detection belongs to. There is no way to determine if two frames received at two different scanners are actually the same, as a probe request does not include a sequence number that would permit differentiating between two separate such frames from the same device.

**MAC address issues** There used to be a time when a MAC address could be more or less used as a stable and unique identifier for a device. This is no longer a justifiable assumption. Some devices change their MAC address seemingly at random, as also reported by Musa and Eriksson (2012a). This is known in particular in the case of some Apple devices Stites and Skinner (2014). Perhaps even worse when using a MAC address for device identification, is that we have noticed cases where different devices use the *same* MAC address.

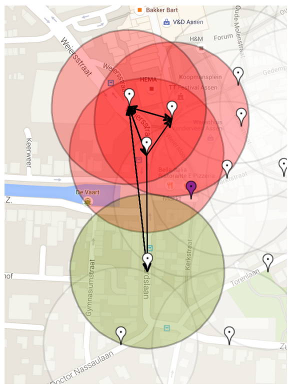


**Figure .2:** Movement path of static device (the circles are 100m visual guides, they **do not** represent the cover radius)

**Correct inference about the population** For various types of applications with societal relevant impacts, it is necessary the correct value about the number of poeple is estimated. Estimation of such numbers is yet another challenge to overcome due to the existence of randomize MAC addresses, people with multiple wifi enables devices, or people with no devices. These challenges are also alleviated when the number of people with more than one device becomes specific to spaces or occasions. For instance, when students of a course all need to bring a laptop for only one session.

**Lack of coordination** Because there is no coordination between devices and scanners, no ideal probe transmission rate can be determined or let alone set. We have witnessed a huge variation in transmission rates, caused by seemingly random behavior when a device switches its WiFi module on or off. This behavior is also dependent on the device, as reported in Cunche (2014) where a comparison between Apple

**Table .1:** Sources of noise in set of detections

| **Description** | **Source** | | | **Correctable?** |
|---|---|---|---|---|
| | **Scanner** | **Device** | **Method** | |
| Faulty scanner | Yes | No | No | Easy |
| Dynamic, irregular ranges | Yes | Yes | No | Hard |
| RSSI issues | Yes | No | Yes | Hard |
| Timing errors | Yes | Yes | Yes | Hard |
| Multiple addr. per device | No | Yes | No | Easy |
| Multiple devices per addr. | No | Yes | No | Easy |
| Uncoordinated probes | No | Yes | No | Medium |
| Lost frames | No | No | Yes | Medium |

and Samsung devices is presented. As a result, the effect, in combination with the unreliability of the wireless medium, is a data set with detections that can make the movement of a device seem mostly erratic. To illustrate what this may lead to, consider Figure .2, which is taken from one of our measurements. In this case we have a known, non-mobile device appearing as a device that moves in a loop, with random frequency and random speed. An actual mobile device could exhibit an even more chaotic behavior. Instead of moving in what would be a straight line, the detections would show it moving in small irregular circles while eventually getting closer to its destination.

By-and-large, there are many sources that introduce *noise* into a set of detections. Table .1 lists the main sources of errors that introduce seemingly chaotic behaviors in datasets collected through wifi scanning. The last column notes the levels of difficulty in dealing with these problems as experienced by us. The ones marked as hard could even be impossible to fix.

## 2.3 Alternatives

Besides using WiFi, there has also been considerable research in using other techniques. In the following, we briefly mention the most prevalent ones.

**CCTV** Classically, data from crowds was obtained using visual systems. These systems are known for being able to infer information on crowds such as congestions, overcrowding and blocking (Siebel and Maybank 2004). However, systems that use visual data are known not to scale well: First of all there is a high cost in deploying a large number of video cameras; secondly the data itself is difficult to extract and process. These systems have an important advantage of being simple to check, errors in data extraction can be corrected by manually checking video logs, but this is a timely and costly procedure.

**Bluetooth** Another alternative to using WiFi is Bluetooth (or other RF based communication technologies such as NFC, RFID, ZigBee, etc). Bluetooth uses the same protocol (similar to WiFi probe requests), for establishing a connection. Such messages contain the Bluetooth MAC address, which again can, in principle, be used for device identification. An important potential advantage of Bluetooth is its more restricted transmission range. As a consequence, provided enough scanners have been positioned, we can obtain a more accurate account of trajectories.

Unfortunately, and as also reported by Schauer et al. (2014) and others, coverage by Bluetooth devices is very low. In practice, many more devices have WiFi enabled than Bluetooth, rendering Bluetooth-based tracking ineffective for many situations. It is unclear if this situation will change in the near future, although wearbles that connect to smartphones through Bluetooth may change this situation.

**Active badges** A very different approach is to use active badges. In general, an active badge is a proprietary device that transmits and receives beacons to other devices and possibly also badges. An important aspect of active badges is that they can be worn in such a way that the *direction* in which a person is facing can be reliably determined. This is caused by the fact that the human body obstructs many transmissions so that we get the same effect as using a directional antenna. Using this feature, studies have been recently carried out to not only follow the trajectories of visitors at an exhibition, but to also determine at which exhibit they were looking (Martella et al. 2016). Using active badges is not efficient in the case of events with large number of participants. Another drawback of these devices is their dependence on extra equipment being carried by people.

**GPS-based systems** The most common method to identify trajectories of individuals is given by the Global Positioning System (GPS). However the GPS receiver has an error of more than a few meters. There are numerous sources for these errors and they are analyzed in Evans et al. (2002). The accuracy of these systems is also dependent on the speed of the device as shown in Aughey and Falloon (2010). They are only meant for outdoor purposes and even when direct line of sight to the positioning satellites is hindered location data cannot be provided (e.g. existence of a forest canopy) DeCesare et al. (2005). These problems are not local to GPS but extend to other specialized positioning systems such as the Argos satellite system Jonsen et al. (2005); Baratchi et al. (2013) used to track animals.

In Thiagarajan et al. (2009) the authors present a way of smoothing the path taken by an individual, as given by raw GPS data. The examples they show present a data set where multiple consecutive detections move back and forth, circling the street the individual is on. This behavior is similar to the behavior we present in this article. Yet the technologies and methods used are different. To extract a clear path the authors use outlier removal with interpolation followed by Viterbi matching. Similarly Yan et al. (2010) use outlier removal and Gaussian kernel regression to smooth the paths shown by GPS data. Their methods are not directly applicable to our scenario. Data collected using GPS has a finer location accuracy than the data from wifi scanners. In contrast, we have low number of detections and a rough approximation (in the order of hundreds of meters) of the actual position. While precise location of a device is a positive attribute of GPS systems, it is invading the privacy of the users carrying GPS equipped devices. In many civil applications based on GPS such accuracy is not required. Thereby, many GPS based data analysis is done by descritizing GPS data to a courser grained accuracy level such that the privacy of the users is not invaded (Baratchi et al. 2014).

**Systems based on regular cell-phone traffic** Finally, it is has become common to use traditional traffic generated by cell phones to locate and track people. Straightforward is to use the identifier of the cell to which the phone is connected as the base for location. However, considering that cells are relatively large, sometimes having a diameter of even a few kilometers in rural areas, these cell-identification sensing techniques are far from accurate. As explained by Ficek et al. (2013), various additional methods and techniques are necessary to improve accuracy before being able to come to any sensible tracking information.

Over all these methods WiFi does have a few clear advantages. It takes advantage of the fact that smartphones, equipped with WiFi modules are ubiquitous. This means that the cost of deploying a WiFi crowd tracking system is small and that there is no need for the cooperation of the individuals in the crowds being tracked. Furthermore, the data outputted by the system is easy to process and contains no false positives. It also has a high accuracy compared to cell-phone communication.

## 3 Handling raw sensory input

The basic concept of scanning for WiFi frames and being able to track crowds is simple. The required hardware for scanning, a WiFi enabled device, is easily accessible and cheap. This hardware does require special software in order to enable the scanning functionality.

To offer a concrete example, for our research we primarily use a platform consisting of scanners (BlueMark 1000 series) and a server to store the data trace. For communication and data transfer, we use broadband communication (a 4G mobile network). The Bluemark BM1000 scanner has 32 MB RAM, 8 MB flash and an 384 MHz CPU. It runs openWRT [1] as its operating system, together with an application, developed by us, that collects the required data. The scanner uses a directional antenna with an antenna gain of around 12 dBi. Like the case with any wireless technology, the shape of the area in which WiFi frames can be received is irregular and inconsistent in time.

The scanner outputs data in SQL text format, it compresses it and periodically sends it to the server. At the server the output files are decompressed and set for long-term storage and analysis. Scanners are synchronized using NTP and reboot daily at 5AM. This last step is meant to minimize errors.

In order to build a scanner application we used the libpcap library [2] to gather WiFi frames and an MD5 Rivest (1992) library to hash device identifiers in order to preserve the privacy of pedestrians.

### 3.1 The 802.11 protocol family

Commonly known as WiFi, the IEEE 802.11 family of protocols [3] standard defines medium access layer and physical layer specifications. At the medium access layer the transmitted data is split in frames. The standard defines 39 frame types and sub-types as well as a number of reserved ones. All the frames contain a header with information relevant to the connection itself. Even if the connection itself is encrypted, the header is sent in clear. The header may include identifiers for senders and receivers, called MAC addresses. The MAC address is meant to **uniquely** identify a wireless device. IANA [4] provides OUI (Organizationally Unique Identifier) numbers for hardware manufacturers and hardware manufacturers use these numbers to generate MAC addresses. The OUI is used as the first 24-bits of the MAC address. Because a device can be uniquely identified, detections can be correlated across multiple scanners.

Figure .3 is a diagram describing the general frame format. The minimal frame format contains only the Frame control, Duration/ID, Address 1 and FCS fields. It is common that the first three MAC addresses to represent the source address (SA), destination address(DA) and the basic service set identifier (BSSID) which identifies the network. There are frame types that do not contain a source address, for instance, the Clear To Send (CTS) frame, used to signal that there are no other transmissions taking place. Out of the 39

[1] https://openwrt.org/

[2] http://www.tcpdump.org/

[3] http://standards.ieee.org/getieee802/download/802.11-2012.pdf

[4] http://www.iana.org/

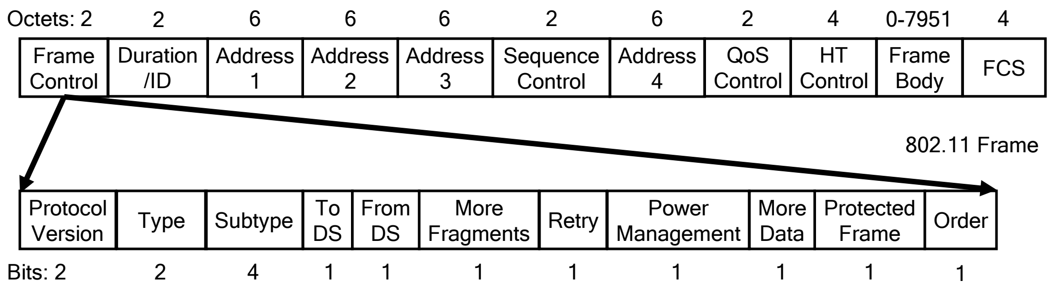


**Figure .3:** WiFi, 802.11 General Frame Format

**Table .2:** 27 frame types/sub-types that contain a source address

| **Type** | **Sub-type** | | |
|---|---|---|---|
| Data | Data | Data+CF-ack | Data+CF-poll |
| | Data+CF-ack+CF-poll | Null | CF-ack |
| | CF-poll | CF-ack+CF-poll | QoS Data |
| | QoS Data+CF-ack | QoS Data+CF-poll | QoS Data+CF-ack+CF-poll |
| | QoS Null | QoS+CF-ack(no data) | QoS+CF-poll(no data) |
| Management | Association_Request | Reassociation_Request | Probe_Request |
| | ATIM | Disassociation | Authentication |
| | Deauthentication | Action | |
| Control | Block_Ack_Request | Block_Ack | PS_Poll |
| | RTS | | |

frame sub-types only 27 have a source address. Table .2 lists the 27 frames.

WiFi uses a frequency band that is not common in nature. This is especially true considering that according to the standard data is encoded in a digital form, also uncommon in nature. Furthermore, all 802.11 frames contain a Cyclic Redundant Check (CRC) number in the FCS field. The CRC is used to identify transmission errors. If the CRC is missing or does not match the expected value, the frame is automatically dropped by the kernel. All these features of the WiFi protocols guarantee that a scanner can never detect a device that does not exist. In other words, there are no false positives during normal usage.

A malicious device can change its MAC address and even send frames having the MAC address of other devices as the source address. This would be an example of false positives. In the previous section we discussed the case of Apple devices which behave in this manner Stites and Skinner (2014). However, this type of detections can be easily identified and filtered (more details in section 3.2). However, once a device is connected to a network it needs to use the same MAC address. This means that a device cannot always hide itself from the WiFi tracking. For the end user one way of preserving privacy is to keep the WiFi module offline unless in use. Most Operating Systems leave the WiFi module on by default, and so do most users. There are also convenience and usability reasons that would motivate the continuous use of WiFi.

The Frame Body field represents the payload, the useful data of a transmission, usually in the form of IP packets. This part of the frame may be encrypted if the devices are connected to a network with WEP or WPA. Scanners should completely ignore the payload even when it is sent in clear text. Scanning or recording payload data raises important privacy issues. The only relevant data for WiFi tracking is the MAC address of the transmitter, the source address.

**Table .3:** All vs *Probe_Request* frames

| | **High Traffic Channel** | **Low Traffic Channel** |
|---|---|---|
| **# All frames** | 2906574 | 18936 |
| **# Probe_Request frames** | 10484 | 12404 |

Generally WiFi networks have one hotspot (a WiFi router) and multiple mobile devices that connect to it. The hotspot advertises its SSID (name of the network) and other information such as accepted speeds using *Beacon* frames. Mobile devices listen for *Beacon* frames and connect to the network, if it is known or after they request user input.

According to the 802.11 protocol family, mobile devices can actively scan for networks using *Probe_Request* frames. This permits a mobile device to find and connect to a hidden network, one that does not send *Beacon* frames. A *Probe_Request* is sent on every channel, once for each recently connected network. A *Probe_Request* may contain an SSID for one of the already known networks. The hotspot responds with a *Probe_Response* frame, containing data about the network capabilities. After this the mobile device and hotspot can start the process of association, which in the cases of encrypted networks is followed by authentication. If these steps are successful, then the mobile device is connected to the network and can start communicating.

Roaming represents a mobile's device capability to change the hotspot it is connected to without causing an interruption in service. This is common for mobile phones on the GSM network. A phone call is not interrupted even when moving. In order to enable roaming capabilities *Probe_Request* frames are sent even when a device is already connected to a network. This way a device can identify other hotspots with better signal and can dynamically change to them. Because they are sent at somewhat regular intervals as long as the WiFi module is on, most experiments are run filtering everything but *Probe_Request* frames.

The frequency with which *Probe_Request* frames are sent is not clearly defined and in practice it is dependent on multiple factors. Each network stack implementation has different policies and rules on setting the frequency and most commonly it is dependent on battery status. For example, a device with low battery tries to conserve it as much as possible and sets a very low frequency at which to send *Probe_Request* frames. We found it to be common for the frequency of a scan (one frame for every channel and recently connected network combination) to be set to 30 seconds.

In order to confirm the behavior of mobile devices when sending *Probe_Request* frames we conducted a small experiment. We counted the total number of frames and the number of *Probe_Request* frames detected by one of our scanners. The scanner had two antennas set on two different channels and the data was recorded for a period of one day. The channels were chosen to be the ones where we detected the most and respectively the least number of frames for a previous period of one day. The results are presented in Table .3. The number of *Probe_Request* frames are very similar for both channels. In contrast, the total number of frames differs with several orders of magnitude. There are multiple reasons for which the number of *Probe_Request* frames do not perfectly match, the main one being the noise in the medium. It is common for WiFi frames to be lost or dropped because of CRC errors. Because the numbers of *Probe_Request* frames on difference channels are similar, the experiment confirms that in order to scan for available networks a mobile devices sends *Probe_Request* frames on every channel.

After frames are collected by the scanners they need to be converted into detections and sent to the server. A simple approach would be to create a detection for each individual frame. But, not every frame is a good representative of a detection. For instance, it is impossible to create detection from frames that don't contain a source address as there would be no way to set the *device_id*, frames that arrive in a very short interval would generate detections that are indistinguishable or similar enough that they would be irrelevant. In order to clean this raw data set and keep only detections that are relevant we propose a number of filters, some at the scanner and some other the server level. These filters are described in detail in our previous work Chilipirea et al. (2015a) and we present some of the basic concepts behind them in the next subsections.

## 3.2 Filtering data at the scanners

With multiple scanners and one central server the first issue when trying to achieve scalability of the system is the bandwidth usage. In order to preserve bandwidth as much data as possible needs to be removed at the scanners. Even better: unnecessary detections are discarded and not sent to the server. For instance, frames that do not have a source address need to be ignored.

**Filter 1** removes frames that do not contain a source address address. Only the sub-types from Table .2 can pass the filter. The filter also removes frames that are not generated by a mobile device. Fields $fromDS$ and $toDS$ (DS represents the Distribution System, a wired network) from the 802.11 frame, in Figure .3, indicate if a frame is sent to or from a distribution system. The filter is extremely fast: it only needs to check the type of the frame and in case it is a Data frame it needs to check the $fromDS$ field.

**Filter 2 and 0** are meant to work together. Filter 0 removes all *Beacon* frames but keeps a list with their MAC addresses. These are MAC addresses of WiFi access points or hotspots. Filter 2 uses these addresses to remove all frames sent by hotspots or access points as these are most likely non-mobile devices. These frames are not removed by Filter 1. In our data we encountered frames with $fromDS$ set to 0 while the transmitter address was that of a hotspot.

**Filter 3** is meant to only accept frames that are far enough time wise. The WiFi protocol allows the transmission of multiple frames every second, even by the same device. However, the location of a device does not change significantly in under a second. The filter accepts only frames that have at least three seconds between them. This three second interval was chosen empirically, by looking at the data in our sources. Other values can be chosen depending on the application, for instance, in Musa and Eriksson (2012b), where a similar filter is applied, a one-second interval is used as an aggregation point. This time based filter can also be used to filter devices that are non-mobile: Devices that are detected by the same scanner for several hours in a row. In order to implement this filter there are some memory usage considerations. The filter requires a list with all devices detected as well as the first and last time when they were detected.

In Chilipirea et al. (2015a) we go into details on the performance and the results of each individual filter. To get an idea on the advantages of having these filters at the scanners we present the overall results for them. The filters can be used at the same time and all frames that pass all of them are considered to be a detection and sent to the server. We present the results on two data sets. One data set is generated using a scanner placed inside a room of a student complex of VU University Amsterdam, Netherlands. The other data set is obtained from one of the scanners we used in the city center of Arnhem. Because the two traces were run in different environments, particularities of these environments can be seen in the data. For instance, the Student Complex trace has a lot of data frames that are sent by only two devices, most likely

**Table .4:** Data set characteristics

| | Arnhem | Assen | Student Complex |
|---|---|---|---|
| **# of frames** | 2472380 | 11860349 | 2906574 |
| **# of scanners** | 5 | 27 | 1 |
| **# of days** | 1 | 3 | 1 |
| **Start Date** | 2014-07-07 | 2015-06-25 | 2014-07-04 |

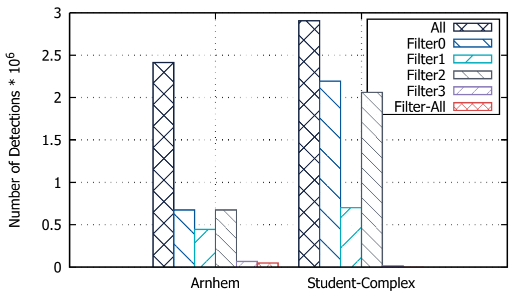


**Figure .4:** Scanner Filters

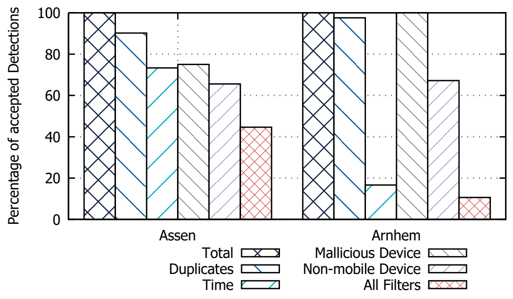


**Figure .5:** Server Filters

belonging to the students closer to the scanner. Table .4 contains the number of frames and the dates on which the trace started.

When testing the filters we initially recorded all frames detected by the scanners and run each filter offline. We did not add any time constraint limitations. We left the software to process the data as fast as possible. This means that we analyzed a few days of data in under 3 minutes, and this has some significant effects on the effectiveness of the 3rd filter. On a real application the filters would run real-time on the scanner.

Filter 1 alone removes about 80% of the frames. The filter is also extremely fast. Filters 0 and 2 are very dependent on the environment. When there are a lot of hotspots near the scanners, there are a lot of *Beacon* frames and the filters remove a larger percentage of the recorded frames. The last filter is more aggressive in this scenario then it would be in real life. However, in real life we discovered that a lot of non-mobile devices, such as printers produce a large number of frames which would be removed by this filter. The results are presented in Figure .4. Filter 0, 1, 2 and 3 are tested independently and finally we start all of them simultaneously. It is clear that the combination of all the filters is more effective than each of them used individually. This means that the filters remove different types of frames and that there is no filter which removes all the frames removed by the others.

## 3.3 Filtering data after centralization at server

When the data arrives at the server it has the following format $< scannerid; deviceid; timestamp >$. Here, *scannerid* identifies our scanner for which the physical location is known, and *deviceid* identifies a mobile device. For privacy reasons, it should be stored as an MD5 hash. Finally, the *timestamp* represents the date and time of the detection, when the 802.11 frame was received at the scanner. We found that this format is somewhat similar across most WiFi tracking projects.

After the data arrives at the server it can further be filtered to help with performing analysis for gaining insight on peoples movements. Some of the filters are:

**Duplicates filter** removes all detections that have the same values for all three, *scannerid*, *deviceid* and *timestamp*. Having these duplicate detections is possible because of clock synchronization errors and buffering between the server and the scanners.

**Time filter** removes all the detections that are not part of the interest period. It is common that data is generated in the testing phase or outside of the area of interest in time.

**The Malicious device filter** is used to remove all detections that have a randomly generated MAC address. This is especially true for Apple devices as advertised in Stites and Skinner (2014). Some Apple devices randomize their MAC address when sending *Probe_Requests* frames. Because the address keeps changing a device using this feature cannot be tracked over multiple sensors. To apply this filter we require that the *deviceid* contains the OUI, alongside the hash of the entire MAC address.

**Non-mobile device filter** removes all detections of devices seen at only one sensor. These devices can appear to be mobile devices such as phones or laptop, but are only used in a non-mobile fashion. An example would be a laptop that is used as a desktop workstation.

We have conducted several experiments where we gather WiFi header data and use it in order to understand pedestrian dynamics. One of our early experiments consisted of only one scanner and was performed inside of a student campus building. We later performed larger experiments which cover entire city centers and were performed during a living statues festival [5] in the city of Arnhem, Netherlands and the tt music festival [6] around the MotoGP event in Assen, Netherlands. These 2 festivals each attracted about a hundred thousand visitors to these two cities. The characteristics of these data sets are presented in Table .4.

By applying these filters we obtained the results in Figure .5. Each filter removes a number of detection from the data sets and by applying all four of them the data sets are reduced even further. After applying the four filters the Arnhem data set has been reduced to 10% of its original size and the Assen one to 44%.

## 3.4 Resulting data set

After the filters have been applied the remaining data is ready for analysis, as it contains far more accurate about the actual movements of pedestrians across those detections. We extracted one day from the Arnhem data set in order to give an example of what can be expected from a WiFi tracking data set.

One of the interesting features to observe in the data is the variation on the number of devices and detections throughout the day. In the data set a device is represented by its *ID* (OUI and MD5 hash of the MAC address) and by all the triplets $< scannerid; deviceid; timestamp >$ that have the $deviceid = ID$. The variation in the number of devices and detections during a day can be observed in Figure .6. Both the number of devices and detections increase during the day and drop during the night. This creates a day/night cycle which is common in most data sets related to human activities. The two patterns created by these numbers do not perfectly match. This is because different devices do not generate the same number of detections.

[5] http://www.worldlivingstatues.nl/

[6] http://www.ttfestival.nl/

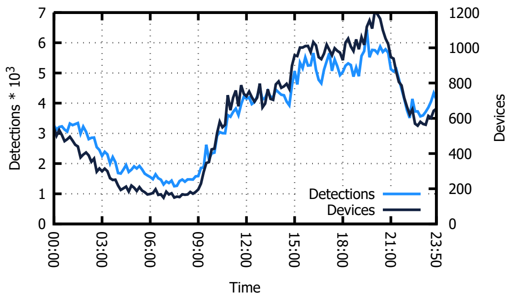


**Figure .6:** # of Detections/Devices over time

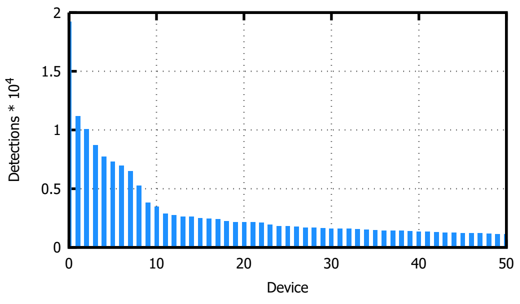


**Figure .7:** # of Detections, first 50 devices

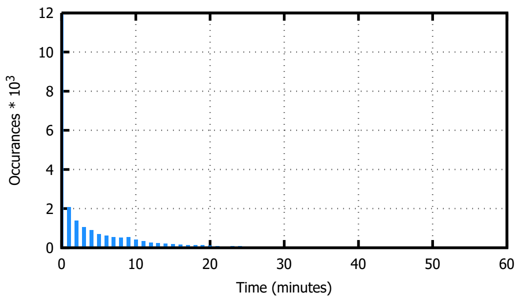


**Figure .8:** # of occurrences of different time periods for which a device is in range of at least 1 scanner

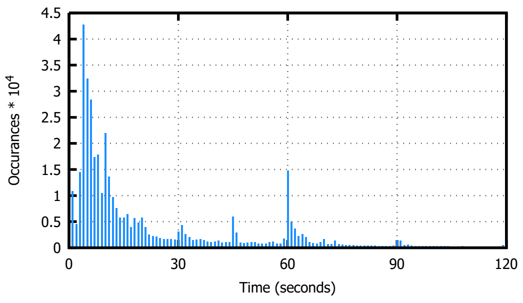


**Figure .9:** # of occurrences of different time periods between two consecutive detections of the same device

The number of detections is dependent on the mobile device usage and the probability of receiving a correct frame given the noise in the environment on the WiFi transmission frequency. To show this we counted the number of detections for each individual device. In Figure .7 we present the results. To improve visibility we ordered the devices in decreasing order of the number of detections they had and we removed the ones with most detections as well as the long tail of devices with few or only one connection. The result can be approximated to a Zipfian distribution Zipf (1949). Many types of data sets studied in social sciences can be approximated to match a Zipfian distribution.

The Zipfian distribution is also visible in other features of a WiFi crowd tracking data set. For instance when we measure the amount of time a device is visible by any of our scanners. Figure .8 represents periods of time for which a device is detected by scanners ordered decreasingly by the number of occurrences for each time value, in minutes. Again we removed the large number of occurrences that were detected only briefly (under one minute) and the long tail of devices that were detected for variously long periods on time. To measure the amount of time in which a device is visible we consider consider the difference between the first and last detections of a continuous set of detections that have no more than five minutes in between.

The quality of pedestrian tracking data sets is given by the frequency with which detections are registered. Having only the detections data set, if there are only few detections of a device the path taken by the device through the city could be impossible to determine. In the case of WiFi tracking this frequency cannot be controlled and is dependent on each of the mobile devices that are being tracked. We measured the frequency for our data set and counted the number of occurrences of consecutive detections of a device, at any scanner, with a fixed number of seconds between them. In Figure .9 we vary the number of seconds between consecutive detections, and show the number of occurrences for each case.

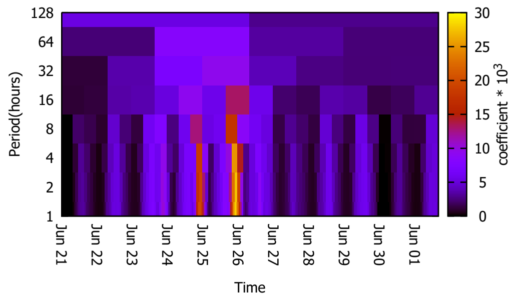


**Figure .10:** Multiresolution analysis of number of visitors in the city during a period of 10 days.

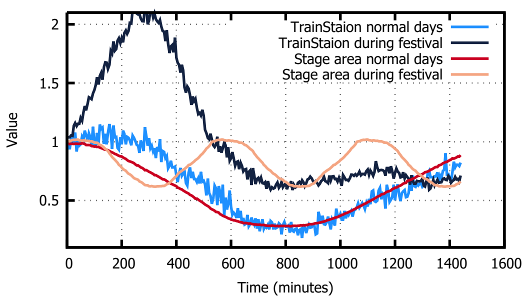


**Figure .11:** Cross-correlation between people coming and leaving stages.

The highest value in Figure .9, at three seconds is most likely caused by the Filter 3 on the scanners. The filter only accepts detections if they are more than three seconds apart. Other interesting peaks are at 30, 60 and 120 seconds. These peaks are most likely caused by the frequency at which most mobile devices send *Probe_Request* frames. We also noticed this 30 second frequency, along with its multiples, on the mobile device we used to initially test the scanners with.

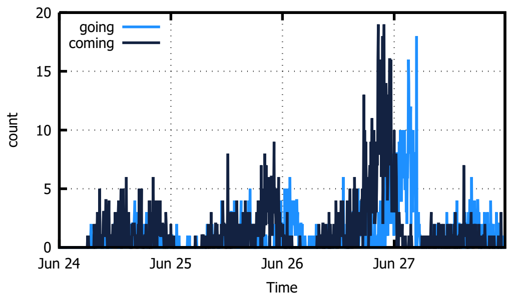
The feature we described above are common for all the data sets we encountered. We believe that having a WiFi tracking data set that contradicts these features could be a sign of errors.

## 4 Data analysis

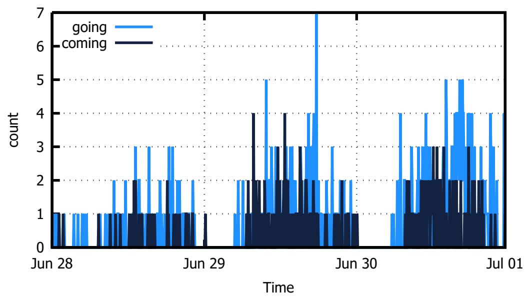
After pre-processing and cleaning the data there still remains the question of how to extract useful information from it. In this section, we give some examples of how such data can be used for this purpose. The dataset that we have collected, contains probe requests from a period where normal activities in a city is changed due to a festival. An example of the type of information that can be extracted is the change of normal usage patterns of space during such an event.

One of the approaches that can be used to acquire a general understanding of how these changes occur is performing multi-resolution analysis. Wavelet transformation is well-known for representing changes in a time series in different resolutions. For our specific dataset, we use a Haar wavelet to see how the changes in city mobility patterns are reflected in the number of people near scanners. When applied to the time series of a specific period of time, the Haar Wavelet can represent the changes between two consecutive timestamps of different duration (increase or decrease in the number of probes). Figure .10 represents the wavelet coefficients over the above-mentioned period of time. Each bar in the figure represents the fluctuations in the number of visitors within a window of time to the next (with the size of a specific window size shown on the y-axis). As seen, even without going in more detail it is apparent that the coefficients appear to be brighter during the three days of the festival (24-26 June) compared to the normal "rhythm" of the city. It is also seen that outside the festival time, these fluctuations in all periods of different length have a repetitive pattern.

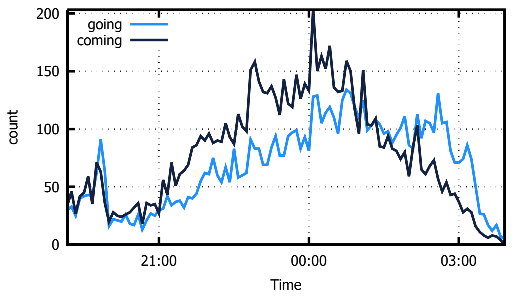
What is shown above is an example of a general analysis but it is also interesting to know how these changes appear at different locations. Each of the spaces covered by a scanner has general use cases during normal days. During the festival time these use cases change. For instance, empty squares turn into stages where music is played, parks turn into places where people camp, and new means of public transport are



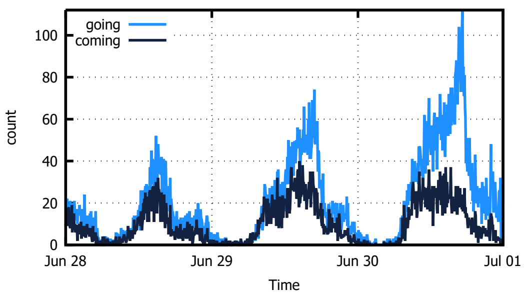
**(a)** Train station - During Festival



**(b)** Train station - Outside Festival



**(c)** Stage Area - During Festival



**(d)** Stage Area - Outside Festival

**Figure .12:** Comparisons of people going to and leaving a stage area and train station during and after festival.

being used. Automatically understanding different semantics and extracting those semantics from probe requests is a challenging problem. Inferring such information requires extensive spatio-temporal analysis. Each of the spaces will potentially represent a different usage pattern, specific to that place. In what follows we take two of the locations covered by a scanner (a train station and a square, respectively) and represent how different spatio-temporal features extracted from the probe requests change over the festival and normal days.

**Incoming and outgoing traffic:** One of the features that changes during the festival is the pattern of incoming and outgoing traffic to a space. In other words, how people go to a place and leave it. Figure .12 compares the train-station and the stage area in terms of this feature. It is seen that there is a delay between the population who come to and leave these spaces during the festival. To better understand the amount of delay between the incoming and outgoing traffic, a cross-correlation function can be used. In general, the cross-correlation graph shows how two correlated time-series follow each other. The first peak in the cross-correlation graph can represent the delay between the two time series. Figure .11 shows the results of applying this function on the incoming and outgoing time series. As seen, during the festival there is a four-hour delay between people who come to and leave the city. This can be related to the duration of stay of visitors in the city. On normal days, however, this delay is about 1 hour which might represent the delay between the appearance of people who leave the city and those who travel to the city for work. On normal days, there is no specific delay between people who come to a stage area and later leave. However, during the stage time there is a 50 minutes delay. This delay may represent the average time visitors stay near a stage for watching the program.

**Group sizes:** Another feature that is interesting to look at are group sizes. Each spot within the city attracts groups of different sizes. It is possible that more groups are formed during the festival. This could

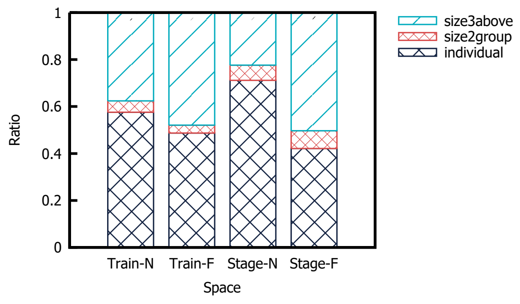


**Figure .13:** Stay duration distribution of different group sizes

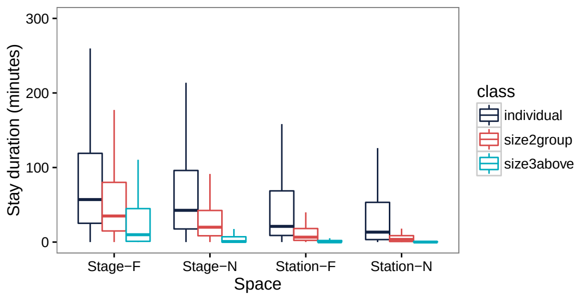


**Figure .14:** Ratio of visitors in different group sizes

be groups of people who know each other and attend an activity together, but also people who temporarily form a group by performing the same activity (over a period of time). Groups can be identified by their synchronous appearance and disappearance near a scanner. As shown in Figure .13 the number of people in groups of larger than three increases during the festival.

**Duration of visits:** Apart from the group size itself, it is also possible that the duration of time people spend near each scanner also changes. As a stage has a program with a specific duration running, it is possible that this also appears as a difference. Figure .14 compares the duration of visits of different group sizes during the festival and normal days. As expected groups of different sizes stay considerably longer in comparison to normal days when looking at stages, but also the train station.

## 5 Toward large-scale crowd-tracking systems

Crowd-tracking systems are used for analyzing or monitoring human activities, such as the flow of pedestrians through busy city centers or the density of individuals in an area, with applicability in a vast number of fields. Examples can span from infrastructure improvement and management to monitoring and security.

Having a scanning system the size of a city raises important issues. Let’s compare it to a one-building scanner system. It is obvious that the area of a building is far smaller than that of a city. But, there are other differences as well. The number of people carrying WiFi devices in a city can be in the order of millions and smartphones and tablets are becoming more and more popular. Covering the area of a city requires a larger number of scanners than a building and, when high accuracy is required, the number of scanners grows even more in order to provide an appropriate granularity. Another point of difference is that moving from a room to another can be achieved by means of a small number of potential paths. However, cities, with their vast networks of streets and pathways, offer a significantly larger number of options to reach any location.

All these problems increase the difficulty in scaling a WiFi tracking system to the size of a city. In order to obtain a high resolution or accuracy on the measurements one possible solution would be to install more scanners. But increasing the number of scanners raises even more problems. The bandwidth usage at the server side increases with each extra scanner and data processing becomes more complex. Previously in this chapter we addressed the problem of minimizing bandwidth usage by having filters at the scanners, which permits the deployment of more scanners. It also minimizes the bottleneck at the server. However, with a high enough number of scanners there will be a need to introduce multiple servers and deal with inter-server communication issues. We found that, in our experiments one server was sufficient, even at peak hours.

Once you have a large-scale crowd-tracking system you need to make sense of the incoming data. This usually means flow and density analysis. Interpreting a large volume of data is it's own challenge, but when dealing with a noisy WiFi system, more difficulties appear. When thinking of a tracking system with stationary scanners one may assume that two consecutive detections of the same device at different scanners represent a movement of the pedestrian carrying the device from one scanner to the other. This is not always true. By analyzing our data sets we identified many cases where two consecutive detections of a device are at scanners which are placed at different ends of the city. This is normal if you consider the noise created by high packet loss rates and devices that can have their WiFi turned off temporarily. Even if we can assume that the device moved from one scanner to the other it is not trivial to determine the path that was used.

Even when there is high noise it is reasonable to expect that two scanners which are "close" have more consecutive detections then those that are further. The assumption is simple, when a pedestrian is detected at scanner A it is extremely likely that a following detection would be at a scanner which is placed "close" to scanner A. However, determining which scanners are "close" to each other is not trivial. By "close" we mean that it is more likely for scanners placed on adjacent streets to have consecutive detections of a device than ones placed in other locations. The problem is made even more difficult by the fact that the placement of scanners usually follows the architecture of the city. It is common for a street map to have an irregular shape.

Having the list of "close" scanners allows one to addresses scalability issues in multiple manners: it permits the deployment of smarter scanner-to-scanner aggregation algorithms; it permits faster data processing at the server level; it enables the simplification of path prediction systems.

A WiFi crowd-tracking system can be represented as an undirected graph, $G = (V, E)$, with vertices represented by scanners $V = \{v_i | v_i \text{ is a scanner}\}$ and edges $E = \{e_{ij} = (v_i, v_j) | i < j; v_i, v_j \in V\}$. $G$ is a **Full Mesh**.

In Chilipirea et al. (2015b) we proposed several methods of creating graphs that show the "closeness" of WiFi scanners. The most direct one is created by using data obtained from the scanners themselves. We call it the **Inferred Graph** (IG) and we define it as a weighted, undirected graph with the same number of edges as the full mesh: $G_{ig} = (V, E_{ig})$ with $E_{ig} = \{e_{ij} = (v_i, v_j); w_{ij} \text{ weight of } e_{ij} | i < j; v_i, v_j \in V; w_{ij} = \text{number of unique devices moving from } v_i \text{ to } v_j \text{ or viceversa}\}$. The inferred graph shows which scanners are "closer" but it does not offer a list with scanners that we consider "close" and scanners that we consider "far". In order to accomplish this we need to select a number of edges with the highest values for $w$. When we created Inferred Graphs for our data sets we discovered that, due to the noise and errors of a real WiFi crowd tracking data set, there is no edge in the graph with a weight $w$ of 0. These errors can be caused by frame collision, environmental causes, or even the temporary disabling of the WiFi capability for some devices.The Inferred Graph also can be used to determine popular sequences of scanners, which can be correlated with popular paths in the city.

As an example, Figure .15 represents the full mesh graph created using the scanners we used in Arnhem. The system gathered detections of devices from 5 scanners placed at reasonable distances (between 70m and 300m) from each other. The numbers the ID of the scanner as it was set by our system, positioned on the map at the same position at which the scanners were during the experiment. In table .5 we show the number of movements from one scanner to the other, grouped by device. It is obvious, and the results show

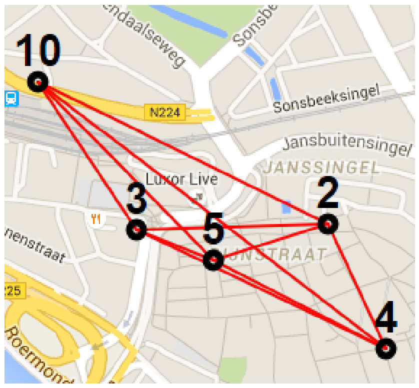


**Figure .15:** Arnhem Scanners Full Mesh

**Table .5:** Arnhem Unique detections for sensor pairs

| $V_i$ | 2 | 2 | 3 | 4 | 2 | 3 | 3 | 2 | 5 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|
| $V_j$ | 3 | 4 | 5 | 5 | 5 | 4 | 10 | 10 | 10 | 10 |
| **# Unique Devices** | 6040 | 3009 | 2331 | 2220 | 1856 | 1604 | 925 | 237 | 92 | 90 |

this, that the scanners that are furthest away, 4 and 10, have the fewest detection pairs. What is interesting is that the ones that are physically closer on the map, 3 and 5, do not have the highest number of devices moving from one to the other. This indicates that the map and people behavior affect the "closeness" of the scanners.

When tracking pedestrians in a city a large number of WiFi scanners is necessary in order to be able to draw valid conclusions about the paths followed, especially when the monitored area has a complex street layout. Increasing the number of scanners inadvertently leads to scalability issues. Even at a coarser granularity the system is still susceptible to scalability problems caused by a high number of detectable devices: poor runtime performance for the data processing algorithms and bandwidth problems. A possible approach for increasing the performance of the system under high load is using a graph for filtering data that indicates movement between scanners that are not "close".

## 6 Conclusions

Tracking pedestrians remains an open problem. There is no solution that offers perfect results in all scenarios. However, WiFi tracking offers one of the most promising alternatives. It takes advantage of how popular smartphones are and how the use of mobile internet is constantly increasing in order to provide tracking data on crowds with large numbers of individuals.

Because smartphone use keeps increasing and there are already multiple alternatives for wearable computing it is correct to assume that the accuracy and the amount of data resulting from WiFi tracking can only increase.

We showed how a WiFi tracking systems can be implemented and what are some of the difficulties of designing and managing such a system on both a small and a large scale. The resulting data can be interpreted using many techniques, a few of which we presented in this chapter.

With many use cases for tracking data, from facility management to simulations that take into account human behavior, interest in the area can only increase. In depth analysis of the resulting data may offer exciting results and opportunities.

## List of acronyms with explanation

- WiFi - wireless fidelity
- MAC - media access control
- SSID - service set identifier
- BSSID - basic service set identifier
- SID - scanner identifier
- MID - mobile device identifier
- T - time
- RSSI - received signal strength indicator
- CCTV - closed-circuit television
- RF - radio frequency
- NFC - near field communication
- RFID - radio-frequency identification
- GPS - global positioning system
- CPU - central processing unit
- RAM - random-access memory
- SQL - structured query language
- NTP - network time protocol
- MD5 - message digest 5
- MHz - megahertz
- MB - megabyte
- IEEE - institute of electrical and electronics engineers
- IANA - internet assigned numbers authority
- OUI - organizationally unique identifier
- FCS - frame control sequence

- SA - source address
- DA - destination address
- CTS - clear to send
- QoS - quality of service
- CRC - cyclic redundant check
- WEP - wired equivalent privacy
- WPA - Wi-Fi protected access
- GSM - global system for mobile communications
- IG - Inferred Graph

## Glossary of terms with explanation

- Detection - a tuple (detection, scanner id, time stamp) that is a record of a device that has passed near to a WiFi scanner
- Inferred graph - graph obtained by using the consecutive detections of devices
- Proximity graph - graph that has as vertices the deployed WiFi scanners and edges between the vertices. An edge exists if two WiFi scanners are reachable by following paths through the city.
- WiFi-enabled device - a device that has WiFi capabilities and is usually carried by an individual
- WiFi scanner - scanner that can send and receive WiFi frames to and from other WiFi-enabled devices

## Index of terms as a list

Please include an index of terms used in the chapter, in a list format.

## References

, 2012: Ieee 802.11: Wireless lan medium access control (mac) and physical layer (phy) specifications. *IEEE-SA*, i –22, doi:10.1109/IEEESTD.2012.6178212.

, 2015: Global internet report 2015. *Internet Society (ISOC)*.

Aughey, R. J. and C. Falloon, 2010: Real-time versus post-game gps data in team sports. *Journal of Science and Medicine in Sport*, **13 (3)**, 348–349.

Baratchi, M., N. Meratnia, P. J. Havinga, A. K. Skidmore, and B. A. Toxopeus, 2013: Sensing solutions for collecting spatio-temporal data for wildlife monitoring applications: a review. *Sensors*, **13 (5)**, 6054–6088.

Baratchi, M., N. Meratnia, P. J. M. Havinga, A. K. Skidmore, and B. A. K. G. Toxopeus, 2014: A hierarchical hidden semi-markov model for modeling mobility data. *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, ACM, New York, NY, USA, 401–412, UbiComp '14, doi:10.1145/2632048.2636068, URL http://doi.acm.org/10.1145/2632048.2636068.

Beard, C. and W. Stallings, 2016: *Wireless Communication Networks and Systems, Global Edition*. Pearson Education Limited, URL https://books.google.nl/books?id=LJ5VCwAAQBAJ.

Chilipirea, C., A.-C. Petre, C. Dobre, and M. van Steen, 2015a: Filters for wi-fi generated crowd movement data. *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, Institute of Electrical & Electronics Engineers (IEEE), doi:10.1109/3pgcic.2015.36, URL http://dx.doi.org/10.1109/3pgcic.2015.36.

Chilipirea, C., A.-C. Petre, C. Dobre, and M. van Steen, 2015b: Proximity graphs for crowd movement sensors. *2015 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC)*, Institute of Electrical & Electronics Engineers (IEEE), doi:10.1109/3pgcic.2015.147, URL http://dx.doi.org/10.1109/3pgcic.2015.147.

Cunche, M., 2014: I know your MAC address: Targeted tracking of individual using Wi-Fi. *Journal of Computer Virology and Hacking Techniques*, **10 (4)**, 219–227.

Curran, K., E. Furey, T. Lunney, J. Santos, D. Woods, and A. McCaughey, 2011: An evaluation of indoor location determination technologies. *J. Locat. Based Serv.*, **5 (2)**, 61–78, doi:10.1080/17489725.2011.562927, URL http://dx.doi.org/10.1080/17489725.2011.562927.

DeCesare, N. J., J. R. Squires, and J. A. Kolbe, 2005: Effect of forest canopy on gps-based movement data. *Wildlife Society Bulletin*, **33 (3)**, 935–941.

Evans, A. G., et al., 2002: The global positioning system geodesy odyssey. *Navigation*, **49 (1)**, 7–33, doi:10.1002/j.2161-4296.2002.tb00252.x, URL http://dx.doi.org/10.1002/j.2161-4296.2002.tb00252.x.

Ficek, M., T. Pop, and L. Kencl, 2013: Active tracking in mobile networks: An in-depth view. *Computer Networks*, **57 (9)**, 1936 – 1954, doi:http://dx.doi.org/10.1016/j.comnet.2013.03.013, URL http://www.sciencedirect.com/science/article/pii/S1389128613000996.

Jonsen, I. D., J. M. Flemming, and R. A. Myers, 2005: Robust state-space modeling of animal movement data. *Ecology*, **86 (11)**, 2874–2880.

Kim, Y., H. Shin, and H. Cha, 2012: Smartphone-based wi-fi pedestrian-tracking system tolerating the rss variance problem. *IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 11–19.

Liu, H., H. Darabi, P. Banerjee, and J. Liu, 2007: Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, **37 (6)**, 1067–1080, doi:10.1109/TSMCC.2007.905750.

Martella, C., A. Miraglia, M. Cattani, and M. van Steen, 2016: Leveraging Proximity Sensing to Mine the Behavior of Museum Visitors. *IEEE International Conference on Pervasive Computing and Communication*, IEEE Computer Society.

Musa, A. and J. Eriksson, 2012a: Tracking unmodified smartphones using wi-fi monitors. *Proceedings of the 10th ACM conference on embedded network sensor systems*, 281–294.

Musa, A. and J. Eriksson, 2012b: Tracking unmodified smartphones using wi-fi monitors. *Proceedings of the 10th ACM conference on embedded network sensor systems*, ACM, 281–294.

Rivest, R., 1992: The md5 message-digest algorithm.

Salyers, D. C., A. D. Striegel, and C. Poellabauer, 2008: Wireless reliability: Rethinking 802.11 packet loss. *International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 1–4.

Schauer, L., M. Werner, and P. Marcus, 2014: Estimating Crowd Densities and Pedestrian Flows Using Wi-fi and Bluetooth. *Proceedings of the 11th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), ICST, Brussels, Belgium, Belgium, 171–177, MOBIQUITOUS '14, doi:10.4108/icst.mobiquitous.2014.257870, URL http://dx.doi.org/10.4108/icst.mobiquitous.2014.257870.

Siebel, N. T. and S. Maybank, 2004: The advisor visual surveillance system. *ECCV 2004 workshop applications of computer vision (ACV)*, Citeseer, Vol. 1.

Stites, D. and K. Skinner, 2014: User privacy on iOS and OS X. *presented in Session 715 of Core OS WWDC14*.

Thiagarajan, A., L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson, 2009: Vtrack: accurate, energy-aware road traffic delay estimation using mobile phones. *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ACM, 85–98.

Wijermans, N., C. Conrado, M. van Steen, J. Li, and C. Martella, 2016: A Landscape of Crowd-management Support: An Integrative Approach. *Safety Science*, **86 (7)**, 142–164.

Yan, Z., C. Parent, S. Spaccapietra, and D. Chakraborty, 2010: A hybrid model and computing platform for spatio-semantic trajectories. *The Semantic Web: Research and Applications*, Springer, 60–75.

Zanca, G., F. Zorzi, A. Zanella, and M. Zorzi, 2008: Experimental Comparison of RSSI-based Localization Algorithms for Indoor Wireless Sensor Networks. *Proceedings of the workshop on Real-world wireless sensor networks*, 1–5.

Zipf, G. K., 1949: Human behavior and the principle of least effort.