SmartITS: Smartphone-based Identification and Tracking using Seamless Indoor-Outdoor Localization

Article · September 2017

DOI: 10.1016/j.jnca.2017.09.003

5 authors, including:

Tarun Kulshrestha
The Hong Kong Polytechnic University
6 PUBLICATIONS 22 CITATIONS
SEE PROFILE

Divya Saxena
Indian Institute of Technology Roorkee
25 PUBLICATIONS 274 CITATIONS
SEE PROFILE

Vaskar Raychoudhury
Miami University
66 PUBLICATIONS 622 CITATIONS
SEE PROFILE

Journal of Network and Computer Applications 98 (2017) 97–113

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

SmartITS: Smartphone-based identification and tracking using seamless indoor-outdoor localization

Tarun Kulshrestha[*], Divya Saxena, Rajdeep Niyogi, Vaskar Raychoudhury, Manoj Misra

Indian Institute of Technology (IIT), Roorkee, India

ARTICLE INFO

Keywords:
Localization
GPS-Wi-Fi-Cellular
BLE
GPS
IEEE 802.11
IoT
Wi-Fi
MAC
LBS
Bluetooth
Probe request

ABSTRACT

Localization in both indoor and outdoor environments is a long-studied problem. Using Smartphone for localization has also gained popularity recently. However, none of the existing solutions consider seamless localization and tracking of individuals in both indoor and outdoor stretches with significant accuracy. In this paper, we propose a human identification, monitoring, and location tracking system, called *SmartITS*, which continuously tracks MAC ids of user equipment (Smartphones, BLE tags, and Bluetooth devices) and can provide a Google map-based visualization of their trajectories. Our tracker is a portable mobile entity comprising of a Smartphone and an external Wi-Fi adapter which does not require any extra hardware infrastructure to deploy as well as does not need any modification in hardware design at all. Extensive testing with a prototype testbed system in densely populated areas shows that the *SmartITS* system can seamlessly track user trajectories in indoor and outdoor stretches with a high aggregate location accuracy which is up to 44.49% more accurate than the simple GPS based location tracking system. Our proof-of-concept prototype shows the usability of *SmartITS* architecture. We also perform several experiments for evaluating the Smartphone's performance as a scanner and as a sensor tag.

1. Introduction

In recent years, Smartphones are frequently used for several locations and mobility-aware applications for indoor and outdoor environments. In the race for rapid development of smart cities, Location Based Services (LBS) are used in various applications, such as human movement monitoring, traffic monitoring, crowd-sourcing, place finding, smart driving, and geo-tagging. LBS can play a significant role in making cities smarter by incorporating the knowledge of citizens' whereabouts, their visiting places, and trajectories.

In order to monitor and track the movement patterns of one or more persons in a densely populated area, the persons must be uniquely identified. Most of the mobile sensing applications are based on GPS, which is highly battery-consuming, and its positioning error can vary from 5 to 120 m (or more). However, the GPS accuracy also depends on the quality of Smartphone used. In addition, at many indoor locations, GPS does not work well (Maghdid et al., 2016). Also, many low-end cheap Smartphones have poor GPS accuracy which renders them unsuitable for similar applications. Wi-Fi based localization is an alternate technique for indoor localization which finds the location of a person carrying a Wi-Fi-enabled Smartphone with respect to fixed access points (Waqar et al., 2016). However, the use of Wi-Fi for outdoor positioning is not accurate in compared to GPS. Finally, the Cellular network based localization has an error of more than 300 m. However, accurate positioning can be achieved by using location-fingerprinting and time-difference-of-arrival (TDOA) based on Long-Term Evolution (LTE). The main issues are that these schemes require additional hardware infrastructure for deployment while, accuracy is limited in dense areas (Maghdid et al., 2016).

Existing localization systems (Ren et al., 2016; Im and De, 2016; Raychoudhury et al., 2015) based on the above localization techniques work for individuals and not for a large crowd as a whole. Moreover, they cannot accurately find the location of a person whose trajectory passes seamlessly through indoor and outdoor location trails. Human location tracking/monitoring can allow the authorities to search/identify a lost person among thousands in the crowd, to evacuate people during emergencies, to manage the crowd movements, and to predict the crowd in the future and to plan the resources accordingly. Researchers in (Al-Ali et al., 2008; Mitchell et al., 2013; Mantoro et al., 2011; Mohandes, 2011) have studied the similar type of systems. Al-Ali et al. (2008) proposes RFID-based localization system for Hajj pilgrims where the pilgrim locations are tracked using GPS-enabled RFID readers. Another such system (Mitchell et al., 2013; Mantoro et al., 2011; Mohandes, 2011) use RFID-chip augmented Smartphones to

[*] Corresponding author.
E-mail addresses: tarun.kul@gmail.com (T. Kulshrestha), divya.saxena.2015@ieee.org (D. Saxena).

http://dx.doi.org/10.1016/j.jnca.2017.09.003
Received 2 September 2016; Received in revised form 9 June 2017; Accepted 8 September 2017
Available online 14 September 2017

upload user locations coordinates (along with their identification) to the remote server using available Internet connectivity. Some problems associated with these systems for large crowd localization and tracking are that they are expensive due to the requirement of the high number of RFID readers and infeasible to distribute to a large mass of people unaware of their usage. Moreover, the human body sometimes obstructs active RFID scanning. Also, localization using proprietary products (Libelium, 2016; Cisco Wireless Location Appliance, 2016) pose challenges regarding modification as well as integration with other systems.

In this paper, we take a step forward towards the identification, monitoring, and tracking of individuals in mass gathering using the probe requests emitted from user's wireless devices and a hybrid localization technique. This localization technique is based on the GPS-Wi-Fi-Cellular which tracks people, both in outdoor as well as the indoor environments in a seamless manner. In our proposed system, named *SmartITS*, we use a Smartphone as a sensing unit (called, Portable Sensing Unit (PSU)) which scans the frames transmitted by nearby wireless devices (Wi-Fi / Bluetooth (BT) / Bluetooth Low Energy (BLE) capable) passively and extracts their unique MAC ids in order to localize the carrying owners using the *PSU's* location coordinates. However, to the best of our knowledge, we propose a novel Smartphone-based human identification, monitoring, and location tracking system which does not require any extra hardware infrastructure to deploy as well as does not need any modification in hardware design at all. Moreover, in our system, *PSU*, a sensing unit is enough to capture probe requests and user's location without user's active cooperation.

In summary, the main contributions of this paper are as follows.

- We propose *SmartITS*, a novel identification, monitoring and location tracking system using a hybrid GPS-Wi-Fi-Cellular-based localization technique where the GPS-Wi-Fi-Cellular-enabled Smartphones scan and track wireless devices (Wi-Fi/Bluetooth/BLE) of mobile clients uniquely using their MAC ids. Smartphones used for sensing the users/clients do not need any modification in the hardware and do not require any Internet connection.
- We perform several experiments on Smartphones for evaluating their performance as a scanner and a sensor tag. Extensive simulation using a Wi-Fi frame injector shows that in indoor environment, each *PSU* handles high incoming frame rates (more than 5000 frames per second) and achieves high data upload rate via a lightweight and bandwidth adaptive technique. *SmartITS* also allows users to increase their Smartphones' detection probability (generation of probe requests) during the emergency.
- Real-time testing with a prototype testbed in a combined indoor-outdoor trajectory containing a large number of people and using Smartphones as *PSUs* have corroborated our simulation results and shows that *SmartITS* can achieve up to 44.49% higher localization accuracy w.r.to pure GPS-based system for indoor-outdoor environments.

The rest of this paper is organized as follows. In Section 2, we discuss the background of the wireless probe requests and its structure. Section 3 elaborates the related works. Section 4 gives the description of the system model and system architecture is discussed in Section 5. In Section 6, we describe the *SmartITS* system implementation and its operations. We have conducted several experiments for analyzing the *SmartITS* system performance in Section 7. Finally, we conclude the paper in Section 8 and provide directions for possible future extensions.

2. Background

A Smartphone with the increasing processing power can be used as a portable scanner for sensing nearby Wi-Fi devices. Wireless probes

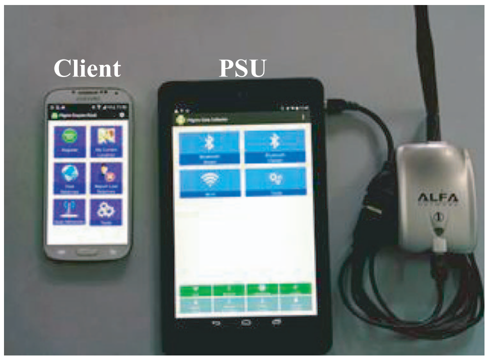


Fig. 1. Smartphone working as a Client (left) and *PSU* (right).

are 802.11 Management frames (IEEE 802 LAN/MAN Standards Committee, 1999) emitted by wireless devices in a pre-connection phase. When the Wi-Fi of Client's Smartphone is ON, it sends probe requests periodically to find an access point (AP) to get associated. Even if it is connected to an *AP*, it will try to find a better *AP*. These probe requests are intercepted passively by *SmartITS* (working in a monitor mode) using external Wi-Fi adapter. Fig. 1 shows a setup of *SmartITS* having a Client and a *PSU*.

The monitor mode allows a *PSU* to capture all the frames on the wireless medium; even the frames are not destined to it. The monitor mode does not require a device to be connected to a network. The promiscuous mode allows a *PSU* to capture only the frames belonging to the network it is connected. In the promiscuous mode, Wi-Fi frames are processed, and the IEEE 802.11 frame header is removed, while in the monitor mode all the frames are passed intact without removing any header. Therefore, it is better to use the Smartphone on monitor mode instead of promiscuous mode. A Smartphone transmits all kinds of frames, but probe request frames are of interest because they are transmitted on all channels and mostly when the Smartphone is not connected to an AP. The frame header contains important fields, such as *frame type*, *subtype*, *transmitter address*, *receiver address*, etc., where *transmitter address* is the MAC id of a user's Smartphone available near to the *PSU*'s scanning range.

In the rest paper, the terms *frame* and *probe request* are used interchangeably.

3. Related works

Nowadays, the usage of Smartphones is continuously increasing all over the globe. Recently, researchers have focused on developing Smartphone-based location-aware human identification, monitoring and tracking applications for both indoor and outdoor environments. The main requirements of a smart human location tracking/monitoring system for both indoor and outdoor environments are – high accuracy, low response time, low cost, high coverage and scalability (Maghdid et al., 2016). Recently, researchers have proposed the use of sensor-enabled Smartphones as a *tag* for large-scale human sensing/tracking which requires special software and hardware that must be incorporated together. Some of the Smartphone-based human/object location tracking systems, such as PDX Bus (Maghdid et al., 2016) requires an application to be installed on the Smartphone.

Many research works are focusing on single positioning/wireless technology, such as RFID (Ni et al., 2011), Wi-Fi enabled devices (Yang et al., 2015), BT/BLE tags (Deepesh et al., 2016), and Smartphone's GPS/General Packet Radio Service (GPRS) (Guido et al., 2013) to track human in indoor and outdoor environments.

RFID-based tracking and monitoring systems have significant performance issues, such as human body interference, expensive RFID readers, and low coverage area when deployed at the densely crowded places. Wi-Fi-based indoor localization, for instance, *fingerprinting* and *ranging*, both use Received Signal Strength Indicator

(RSSI) measurements to estimate users' positions. However, *fingerprinting* methods are not robust to environment changes while *range-based* methods have significant localization errors due to the instability of Wi-Fi signals (Maghdid et al., 2016). Wi-Fi-based indoor localization faces some challenges, e.g., signal multipath/interference caused by the interior setup and blocking signals, and requirement of additional hardware(s) (Mautz, 2009).

BT is one more possible solution for Smartphone-based indoor localization (Fazli et al., 2011) because of its low-cost and low-power consumption. The main issues of not using the *BT* to track the human at large crowd are its high discovery time and low range for scanning. Other *BT*-based indoor localization schemes use BLE tags based on RSSI measurements which have huge location errors due to non-uniform shadowing (Maghdid et al., 2016).

Another possible solution to track the Clients' location in the large crowd is to use GPS sensor of Smartphone and continuously updates the location to the remote server(s) using the Internet. However, it is not feasible that all the Clients in the large crowd will have the Internet connection. Also, GPS does not work well in the indoor, urban and dense area due to the multipath issues (Waadt et al., 2009).

Furthermore, Cellular-based positioning technologies are efficient if more base stations are present near to the localization area. Otherwise, location accuracy is limited. However, the localization scheme based on single wireless technology is not robust because of their trade-off among accuracy, power consumption, and coverage area as well as single point of failure (Boukerche et al., 2008).

To overcome the above-mentioned issues, multiple wireless technologies can be used together, like GPS-Wi-Fi-Cellular to increase localization coverage, and hybrid Global Navigation Satellite System (GNSS) signals along with other wireless/sensor technologies, such as *BT*, Near Field Communication (NFC), acoustic sensors, and inertial sensors to support high accuracy and low space/time complexity (Partyka, 2016). Some existing localization systems, like Wi-Fi-SLAM (Huang et al., 2011), Skyhook and Ekahau (Gallagher et al., 2009) are supporting fair location accuracy (Faragher et al., 2013), while, they require extra hardware infrastructure to deploy, or they need the Internet all the time to locate the Smartphones (Sensewhere, 2016; Zandbergen, 2009).

We, on the other hand, can track, identify, and monitor a Client (indoor and/or outdoor) with no Internet connection in the densely crowded places using Client's Smartphone MAC ids (considering a person is associated/inalienable with his Smartphone) and raw location coordinates. Moreover, *SmartITS* employs Google location API (Making Your App Location-Aware, 2015), a combination of multiple positioning technologies, GPS-Wi-Fi-Cellular and sensor data to achieve the balance between accuracy and power consumption in both indoor and outdoor environments.

Furthermore, many research groups have considerable interest in exploring the usage of Wi-Fi probe requests emitted by user's wireless devices, such as Smartphones and BT/BLE tags. These tiny data packets can be leveraged for the analysis of the multitude of areas, such as social network, human behavior analysis, mobility pattern, traffic control, security, and privacy (Kulshrestha et al., 2017(a); 2017(b)). Moreover, probe requests can be helpful in user tracking and monitoring as well as for discovering nearby Wi-Fi networks. Luzio, et al. presented the idea to collect Wi-Fi probe requests from users' wireless devices, which can be helpful in determining device owner's nature, social information and de-anonymizing the origin of participants and events (Di Luzio et al., 2016).

Rose, et al. proposed a wireless sensor network based approach for exploring the details of wireless access points, monitoring web browsing, Wi-Fi fingerprinting for user's past behavior analysis and Wi-Fi equipped vehicles tracking (Rose and Welsh, 2010). Cunche et al. (2012) analyzed the controlled dataset of Wi-Fi fingerprinting of user's wireless devices for predicting the social connectivity and behavior similarity. Moreover, authors leveraged the intersection among SSIDs of different wireless devices for estimating users' common trajectories, their interests, and similarity metrics. Cheng et al. (2012) extended the previous approaches based on three dimensions: association history, physical closeness, and spatiotemporal behavior. Authors have taken the intersection of captured probe requests over a small sample dataset of users.

Musa and Eriksson (2012) presented a system for tracking Smartphone without installing any app. They use *AP* hardware with custom firmware as a Wi-Fi monitor to capture the frames transmitted from the Smartphones. Authors have presented several techniques for both passive and active tracking. Almost all the Wi-Fi frames contain their unique MAC address which is used for tracking a device. Barbera et al. (2013) collecting the probe requests for finding the social relationships among the Smartphone users. Bonné et al. (2013) proposed a mobile-phone and Raspberry-Pi based system for tracking the people at mass gatherings without user's active cooperation. In most of these approaches, extraction of MAC ids/SSIDs from probe requests requires a two-step process which makes the system complex w.r.to both time and space. One is to capture all probe requests and store them, and later, is to use sniffing tools/procedures to intercept MAC ids from stored probes. While remaining systems require extra hardware for deployment (makes the system complex).

To handle the afore-mentioned problems, we propose, to the best of our knowledge, a novel Smartphone-based human location tracking and monitoring system, named *SmartITS*, which does not require any additional hardware infrastructure to deploy as well as does not need any modification in hardware design at all. Moreover, in our system, a Smartphone used for identification and tracking user's in mass gathering is enough to capture probe requests and user's location without their active cooperation.

4. System model

We design and develop a Smartphone-based user identification and tracking system using the seamless indoor-outdoor localization, named *SmartITS*. *SmartITS* comprises of three main components: a Cloud Server (C_{Server}), a set of portable sensing units acting as a tracker $PSU = \{PSU_1, PSU_2, ..., PSU_N\}$, a set of Clients having either Smartphone or BLE/BT tag $C = \{C_1, C_2, ..., C_M\}$. These components are connected through wireless media.

In this system, *PSUs* capture the frames transmitted on wireless media by *C*, extract the MAC id, store them and send locally processed and filtered data to C_{Server} for further long-term processing. For location identification, *PSU* uses the hybrid GPS-Wi-Fi-Cellular-based localization technique which tracks the individuals seamlessly both in indoor and outdoor environments to achieve high location accuracy. *PSU* uses the *HashMap* data structure (HashMap, 2015) to mitigate the effect of redundant MAC addresses intercepted from the *C* and refreshes them periodically. *PSU* supports delay tolerant networking to cope up with network problems, such as intermittent connectivity, long or variable delay, and asymmetric data rates. C_{Server} can control the *PSU(s)* working if error rate (erroneous data, high sleep time, etc.) is greater than a threshold. Moreover, *C* can install *SmartITS* Client app to register themselves for exploiting better services from C_{Server} such as information of last detected location, tracking of his/her family member(s) in the case of emergency, maximization of their detection probability, etc.

5. System architecture

Fig. 2 shows the system architecture of *SmartITS*. The proposed *SmartITS* system is divided into the Client (C) and *Tracker* sides. A *C* is a person carrying a Wi-Fi/BT enabled Smartphone or a BLE tag whose location is to be tracked. The *Tracker* side comprises of a *Portable Sensing Unit* (PSU) and a C_{Server}. *Cs* moving through the range of a *PSU* will be identified uniquely through their MAC ids and will be associated

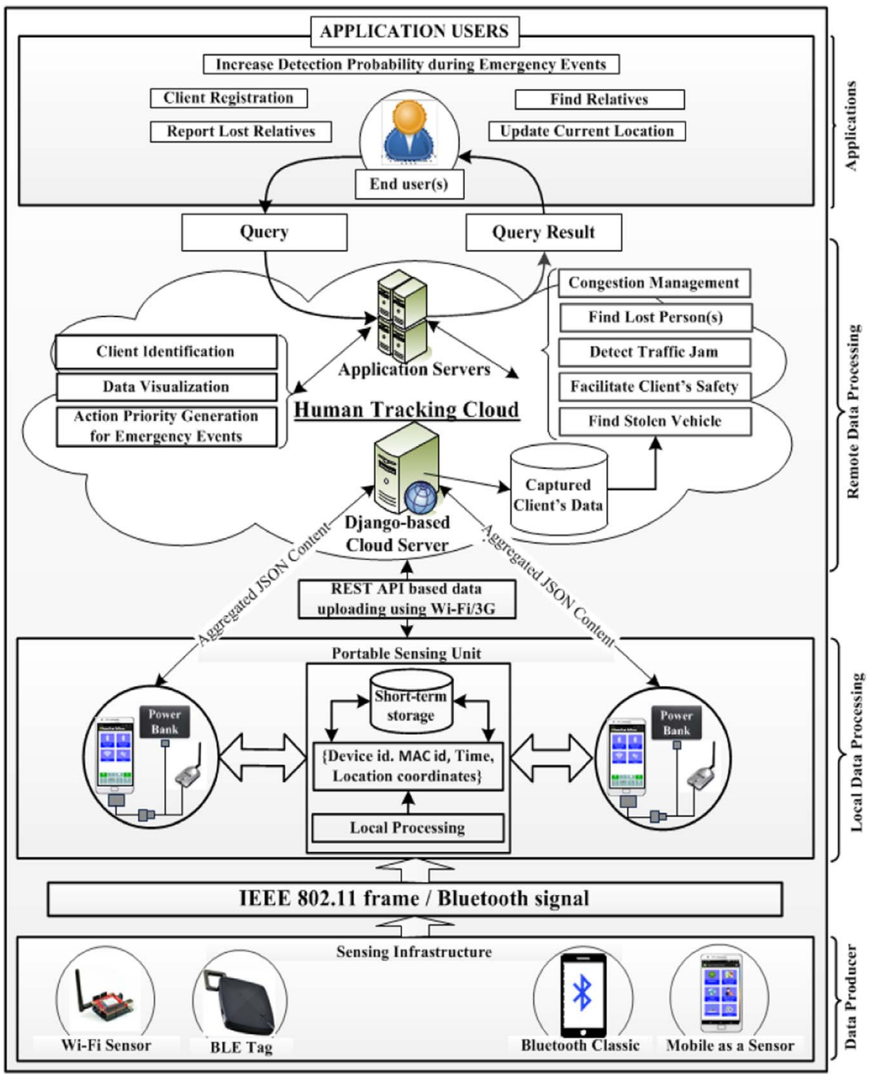


Fig. 2. *SmartITS* architecture.

with the GPS coordinates of the *PSU*. Below, we describe the *C* and *Tracker* modules in details.

5.1. Client side

C devices are Smartphones having Wi-Fi and/or *BT* support. If *Cs* are without Smartphones, the congregation authority can distribute BLE tags (see Fig. 3(a) and (b)). The *C* Smartphone can download an Android Client app to register itself in the C_{Server}. Otherwise, the BLE tags (along with *C* names) are already registered in the C_{Server} before being handed over to the *C*. The Client app helps members of a large group to locate each other and to lodge missing reports.

In this app, at the very first stage, *C*s should register themselves (their MAC ids) to the C_{Server}. *C* can make a query directly to the C_{Server} about their current location. A *C* can also ask to C_{Server} about his lost relatives' current location during an emergency. Client app also has the provision to increase the number of probe requests generation for increasing the detection probability of a *C*.

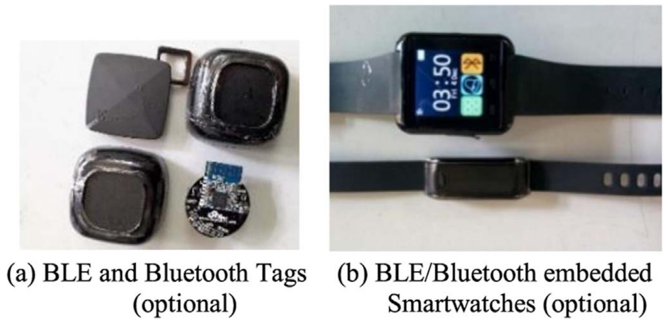

(a) BLE and Bluetooth Tags (optional)
(b) BLE/Bluetooth embedded Smartwatches (optional)

Fig. 3. Client devices in case Smartphones are not available to the *Clients*.

5.2. Tracker side

5.2.1. Portable Sensing Unit (PSU)

The *PSU* consists of a GPS sensor equipped Smartphone, USB Wi-Fi adapter, Android app, and an optional power bank connected through Y-cable. *PSUs* can be static or dynamic/mobile (carried by crowd managing volunteers). The task of *PSU* is to capture the frames transmitted on wireless media by the nearby *C* devices, process the frames to get their MAC addresses (which remains unencrypted even for a secure network), store them locally, and finally, upload the information to the C_{Server}. *PSU* collects data (record) in the following format < *device_id, MAC_address, Time, Client RSSI, Location* > where *device_id* is *PSU* name, *MAC_address* is the MAC id of the *C* detected, *Time* is the physical time at which a *C* is detected, *Client RSSI* is the signal strength of *C* to estimate the physical distance (in meters) between the *PSU* and *C*. *Location* captures the GPS coordinates of the *PSU*. *Location* comprises of *latitude*, *longitude,* and *accuracy*. *PSUs* capture the raw location coordinates of the *C* using the multiple wireless technologies, GPS-Wi-Fi-Cellular (Google Location API) which ensures high location accuracy at the cost of low energy usage.

If two or more *PSUs* are present in each other's transmission range, they coordinate among themselves to divide the channels they want to scan which will distribute the load. Moreover, Internet connectivity available in a single *PSU* can be shared by its neighbors for data uploading.

The detailed implementation and processing of *PSU* are explained in Section 6.

5.2.2. Cloud server (C_{Server})

C_{Server} consists of the back-end data storage unit of the *SmartITS* system. It has three main components, namely: Django-based Web and REST Framework (Django Web Framework, 2015; Django Rest Framework, 2015) to upload and retrieve data efficiently (faster), MySQL database for storing the uploaded data, and Apache Web Server. The data uploaded by the *PSUs* are stored in a time-stamped manner and is used for *C* identification, detection of last location of a lost *C*, location of the *C*'s group/family members, chances of Stampede (too many people jostling in one place), congestion management and traffic (human movement) monitoring. C_{Server} can check the working status of *PSUs* and control their functioning if the error rate in their results is more than a pre-specified threshold.

6. Prototype implementation

This section depicts the detailed implementation of the *SmartITS* testbed system using commercially available hardware and Commercial off-the-shelf (COTS) software modules. Fig. 4 shows the complete data flow of the *SmartITS*.

6.1. Tracker side operation in SmartITS

6.1.1. PSU operation

In our proposed system, the *PSU* performs passive scanning in *monitor* mode where it listens to the channels and intercepts frames/ probe requests generated by *C* devices even the requests are not destined for it. We chose Android as the platform for *PSU* as it is an open source and modern Smartphone's operating system. However, Android does not allow an application running in user space to access a Smartphone's built-in Wi-Fi for discovering nearby Wi-Fi devices. A rooted phone with custom kernel may provide access to built-in Wi-Fi, but still, monitor mode may not be supported by the hardware. *Rooting* is a critical process which voids the warranty and can brick the phone. Also, in this case, there is need to write a separate driver code for every Smartphone with a different chipset. To solve this issue, an external USB Wi-Fi adapter is used with the Smartphone. The driver for the external Wi-Fi adapter is written in user-space using the USB host API which does not require *root* privileges.

Using active scanning may improve the system performance at the cost of higher complexity and separate transmitter and receiver modules which we have decided not to opt for *PSU*. However, we have successfully implemented active scanning at *C* side and optimized the resulting energy usage by adaptively adjusting the probing interval.

A tracker carrying the *PSU* can choose any of the scanning methods: BLE, Bluetooth Classic (BC) or Wi-Fi (Fig. 5(a)). The scanning activity at the *PSU* can be configured using *Setting Activities* (Fig. 5(b)), such as upload data rate (to the C_{Server}), *C* device sampling rate, etc. Fig. 5(c) shows the overview of Wi-Fi scanning activity and the user interface which allows a tracker to start and stop Wi-Fi scanning.

As the scanning tasks are independent and running as a background service, Wi-Fi devices, BC and BLE tags can be detected simultaneously by a single *PSU*. We use two different external Wi-Fi adapters, *Tenda* (Tenda Wireless Adapter, 2015) and *Alfa* (Alfa Wireless Adapter AWUS036H, 2015) for passive scanning by *PSU* as they support monitor mode. The ranges covered by Tenda and Alpha adapters are approximately 10–12 m and 50–60 m, respectively.

Every *BT* device also has a unique MAC id which is sent via a response frame to a scanning device (in case of *PSU*). BLE tags work in the same manner although they have lower energy and incur high cost.

For memory and time efficient *PSU*, it should support the caching of only unique MAC addresses intercepted from the *C* and refreshes them periodically. In Table 1, we have listed the variables used to describe the pseudocode of caching algorithm.

Fig. 6 shows an algorithm for caching of detected MAC_{ids} at the *PSU*. When a new Wi-Fi frame arrives at the *PSU*, its *From_DS* ($From_{DS}^{bit}$) bit is checked. If it is enabled, i.e., it is *AP's* MAC_{id}, then that MAC_{id} is added to Router's HashMap (R_{Hash}^{MAP}) which contains a key-value pair where the key is MAC_{id} and value is the detection time of MAC (MAC_{d_t}). Otherwise, *PSU* stores that MAC_{id} in *C's* HashMap (C_{Hash}^{MAP}) if it does not exist and log in a temporary file along with current timestamp (C_t) and location coordinates. If MAC_{id} already exists in the C_{Hash}^{MAP}, its C_t is updated. If C_t is greater than MAC_{id}'s last saved time (L_t) plus a threshold (Th_t^j), a waiting time is added because a *C* device transmits many frames at a time or a *C* device returns to the nearby region of *PSU*. Due to limited main memory in Smartphones, C_{Hash}^{MAP} is cleaned up to remove old MAC_{id}s at regular interval. The cleanup is triggered whenever the number of entries in C_{Hash}^{MAP} reaches above threshold ($TH.\ size$) or if sufficient time (Th_t^j) has passed after the last cleanup time (L_t^C) where $TH.\ size$ is greater than the number of devices detected at interval Th_t^j.

In particular, the records will be uploaded to the C_{Server} only when their number in the temporary file reaches to *maximum records per file* limit (which can be pre-set through settings). The *duplicate detection* is the time interval during which a *C* device MAC id will be

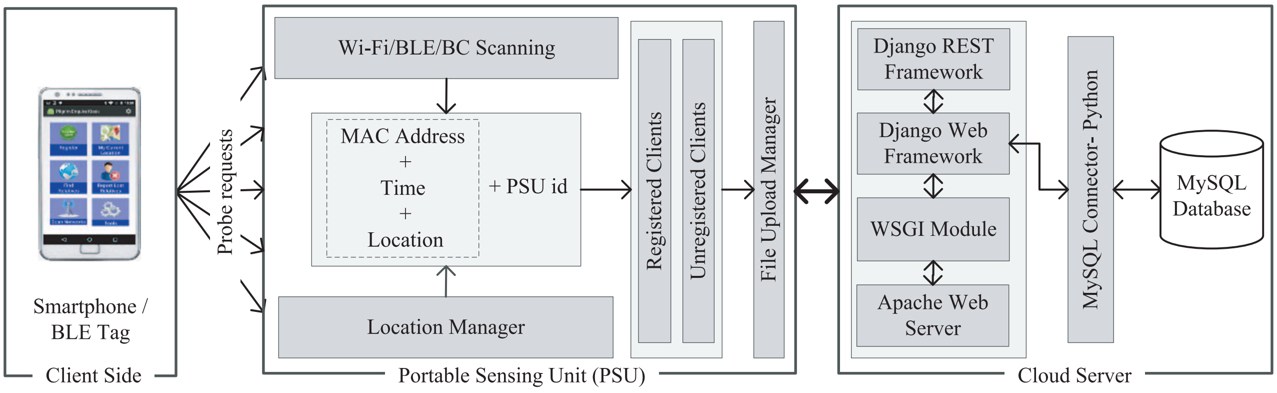
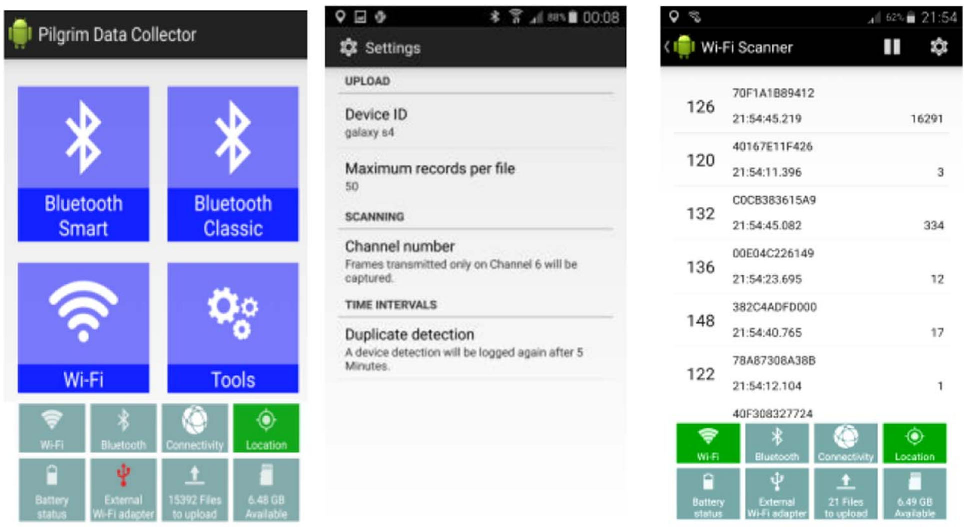


Fig. 4. Data flow among various modules of *SmartITS*.



UPLOAD

Device ID
galaxy s4

Maximum records per file
50

SCANNING

Channel number
Frames transmitted only on Channel 6 will be captured.

TIME INTERVALS

Duplicate detection
A device detection will be logged again after 5 Minutes.

```
126  70F1A1B89412
     21:54:45.219      16291
120  40167E11F426
     21:54:11.396          3
132  C0CB383615A9
     21:54:45.082        334
136  00E04C226149
     21:54:23.695         12
148  382C4ADFD000
     21:54:40.765         17
122  78A87308A38B
     21:54:12.104          1
     40F308327724
```

(a) Main activity (b) Settings activity (c) Wi-Fi scanning UI

Fig. 5. *PSU* application.

Table 1
Variables used in caching algorithm.

Variables	Significance
MAC_{id}	Detected MAC address
C_t	Current time stamp
MAC_{d_t}	Detection time of MAC_{id}
L_t	Last saved time
Th_t^j	Threshold time
L_t^C	Last clean up time
C_{Hash}^{MAP}	Client's HashMap at *PSU*
R_{Hash}^{MAP}	Router's HashMap at PSU
$TH.\ size$	Threshold size of C_{Hash}^{MAP}
$From_{DS}^{bit}$	From DS bit of Frame Control Field (IEEE 802 LAN/MAN Standards Committee, 1999)

```
1.  Received Wi-Fi Frame at PSU
2.  if (Frame's $From_{DS}^{bit}$ == T)
3.      Add $MAC_{id}$ into $R_{Hash}^{MAP}$ <$MAC_{id}$, $MAC_{d\_t}$>
4.  endif
5.  if ($From_{DS}^{bit}$ == F AND $MAC_{id} \notin C_{Hash}^{MAP}$)
6.      Add new $MAC_{id}$ into $C_{Hash}^{MAP}$ with $C_t$
7.  endif
8.  if ($MAC_{id} \in C_{Hash}^{MAP}$)
9.      if ($C_t > L_t + Th_t^j$)
10.         Update $C_t$ of $MAC_{id}$
11.         Save current file
12.         if $C_t > L_t^C + Th_t^j$
13.             Delete old entries from $C_{Hash}^{MAP}$
14.         endif
15.     else
16.         Wait for ($C_t > L_t + Th_t^j$) and repeat steps 8-11
17.     endif
18. endif
19. if $C_{Hash}^{MAP}.size > TH.size$
20.     Delete old entries from $C_{Hash}^{MAP}$
21. endif
```

Fig. 6. Algorithm for caching of MAC ids at PSUs.

logged once only irrespective of the number of times it is detected.

6.1.2. Cloud server (C_{Server})

The C_{Server} provides a RESTful API for uploading or retrieving the data using the Django REST framework. Django implements Python WSGI (WSGI module for Apache, 2015) (as shown in Fig. 4) which is a standard interface between web servers and Python web applications or frameworks. Its primary goal is to provide portability across different web servers. It allows us to choose any web server, e.g., Apache (Apache web server, 2015), Nginx (Nginx web server, 2015), lighttpd (Lighttpd web server, 2015), etc. The C_{Server} uses MySQL database to store the *C* records captured by the *PSUs*.

6.2. Uploading data

The data uploading task is handled by a background service, which uploads files in the time-stamped order of creation. After uploading a file successfully, it is deleted from the internal storage of *PSU*. If uploading fails, it retries to upload file when network connection is available again. The file can be uploaded using either Wi-Fi (default) or mobile data as decided by the Android.

Following three cases are considered for uploading the sensed data:

- *Case 1: Uploading all detected MAC records (except APs)*: In the case of traffic analysis, crowd evacuation analysis and the estimation of congregation strength, the *PSU* uploads every detected MAC record without any restrictions (excluding the MAC ids of *APs*). Therefore, in this case, all registered and unregistered MAC records are uploaded at C_{Server}.
- *Case 2: Uploading the MAC records of BLE tags*: BLE tags having similar predefined prefix/suffix alphanumeric pattern of MAC ids can be scanned and uploaded through *PSU* programmatically after registration.
- *Case 3: Uploading and synchronizing MAC records of registered Clients:* In this case, only registered *Cs'* records are uploaded by the *PSU* to save the Cloud storage, network bandwidth and to provide additional services to registered *Cs*. This is highly useful and required when there is a need for human identification and tracking. This feature optimizes human tracking and energy as previously stored records can be utilized in the future.

In Table 2, listed variables are used to describe the pseudocode of synchronization algorithm.

Fig. 7 shows the algorithm for synchronizing the MAC records (of registered *C*) among *PSUs* and C_{Server} at time interval *t*. C_{Server} synchronizes only registered *Cs* list (R_{list}^C) with the *PSUs'* HashMap, where *PSU* HashMap continuously stores unregistered *Cs* list (U_{list}^P). *PSU* updates its U_{list}^P after synchronization with C_{Server} and uploads only R_{list}^C records to C_{Server}. *PSU* maintains U_{list}^P for some *logging period* (Th_t^l).

Table 2
Variables used in synchronization algorithm.

Variables	Significance
R^{C}_{list}	List of Registered Clients at C_{Server}
U^{P}_{list}	List of Unregistered Clients at PSU
$R^{C\prime}_{list}$	New list of Registered Clients at C_{Server}
C_t	Current Time Stamp
Th^{k}_{t}	Time interval for synchronization
MAC_{d_t}	Detection time of MAC id of UR_C
UR_C	Unregistered Client(s)

```
1.  Synchronize $R^{C}_{list}$ on $C_{Server}$ with PSU at time t
At PSU side
2.  $U^{P}_{list}$ = $U^{P}_{list}$ - $R^{C}_{list}$
3.  Upload $R^{C}_{list}$ records to $C_{Server}$
4.  Wait $Th^{k}_{t}$ for incremental update from $C_{Server}$
5.  $U^{P}_{list}$ = $U^{P}_{list}$ - $R^{C\prime}_{list}$
6.  if $U^{P}_{list}$ updated
7.      Upload $R^{C\prime}_{list}$ records to $C_{Server}$
8.  else
9.      if $C_t > MAC_{d\_t} + Th^{k}_{t}$
10.         Delete $UR_c$ from PSU
11.     else
12.         Wait $Th^{k}_{t}$ for incremental update from $C_{Server}$
13.     endif
14. endif
```

Fig. 7. Algorithm for MACs synchronization between *PSU* and C_{Server}.

During Th^{l}_{t}, C_{Server} updates newly registered *C* list ($R^{C\prime}_{list}$) to *PSU* incrementally at the time interval Th^{k}_{t}. If during Th^{l}_{t}, where $Th^{l}_{t} > Th^{k}_{t}$, condition $C_t > MAC_{d\_t} + Th^{k}_{t}$ is true then unregistered *C*(s) (UR_C) are deleted from *PSU's* U^{P}_{list}, otherwise, wait for Th^{k}_{t} for an incremental update from C_{Server}.

6.3. Client side operation in SmartITS

Client app allows a *C* to register themselves to the C_{Server}. During an emergency, a lost *C* and their relatives can be tracked using the last known location stored at the C_{Server} (see Fig. 8(a)). In Client app, the *C* can update his location at any time as shown in Fig. 8(b). S/he can also increase his detection probability by increasing the generation of probe requests using the transmission time interval (scanning interval) (see Fig. 8(c), by default, transmission interval is set to 10 s).

7. Experimental evaluation

The *SmartITS* system has been developed from scratch, and the application has been extensively tested both in indoor and outdoor environments in lab-based simulation settings as well as using real prototype testbed. In this section, we shall introduce our testbed, dataset, and performance metrics used to test and evaluate the *SmartITS* system.

7.1. Background of simulation experiment

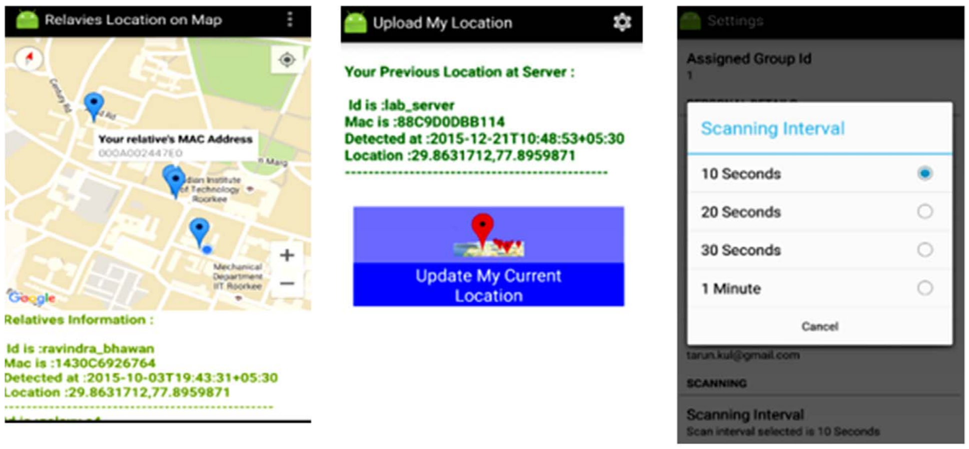
In most of the experiments, we need either to capture or to transmit Wi-Fi frames. Therefore, for simulation purposes, we have developed a Linux-based Wi-Fi frame injector using the PCAP library (TCPDUMP and LIBPCAP, 2015) which is installed on Dell OptiPlex system having 16 GB RAM, Intel Core i7-4790 Processor, and 500 GB HDD. Wi-Fi frame injector works as a *C* to test the processing capability of the *PSU* and C_{Server}. It can adjust the following parameters: *Number of frames* to transmit, the *delay* between the frames, *types of frames* (Data & Request to Send (RTS)), and *MAC id* of *C* (same and / or different). We have used three different smart mobile phones (*Samsung Galaxy S4* (SG_S4), *Dell Venue 8* (DV_8) and *Google Nexus 5* (GN_5)) and a Linux-based desktop computer as *PSUs* to capture the frames transmitted from nearby *C's* devices. Battery capacities of *SG_S4*, *DV_8*, and *GN_5* are 2600 mAh, 4100 mAh and 2300 mAh, respectively. And, the Android version for *SG_S4*, GN_5, and DV_8 are Lollipop (5.0.1), Marshmallow (6.0.1), and Kit Kat (4.4.4), respectively. Table 3 shows the hardware specifications of the Smartphones used as *PSUs*.

There are three external Wi-Fi adapters, *Alfa* AWUS036H (Alfa Wireless Adapter AWUS036H, 2015) (*Alfa_1*, and *Alfa_2*) and *Tenda* W311M (Tenda Wireless Adapter, 2015) for facilitating wireless capabilities to the *PSUs*.

For a period of 8 months (July 2015 – Feb. 2016), we have collected 27,47,767 records with more than 82,550 unique MAC ids. Fig. 9 shows people location traces on the map using Wi-Fi scanning. As already discussed in Section 5.2, each record is composed of the following fields < *device_id, MAC_address, Time, Client RSSI, Location* > . Data collection process is continuous and round-the-clock.

7.2. Background of prototype testbed

PSU is a major component of *SmartITS*, responsible for scanning the *C* devices and uploading the tracking data. Experiments are carried out in and around Indian Institute of Technology, Roorkee (IITR) campus (*Exp@IITR)* and are significantly dense. IITR is an academic


Relavies Location on Map
Your relative's MAC Address
000A002447E0
Indian Institute of Technology Roorkee
Mechanical Department IIT Roorkee
Google
Upload My Location
Update My Current Location
Settings
Scanning Interval
Cancel
SCANNING

Relatives Information :

Id is :ravindra_bhawan
Mac is :1430C6926764
Detected at :2015-10-03T19:43:31+05:30
Location :29.8631712,77.8959871

Your Previous Location at Server :

Id is :lab_server
Mac is :88C9D0DBB114
Detected at :2015-12-21T10:48:53+05:30
Location :29.8631712,77.8959871

Assigned Group Id
1

Scanning Interval: [✓ 10 Seconds] 20 Seconds 30 Seconds 1 Minute

tarun.kul@gmail.com

Scanning Interval
Scan interval selected is 10 Seconds

(a) Client relative's location activity (b) Client current location and update (c) Transmission time settings

Fig. 8. Android-based Client App.

Table 3
Hardware specifications for the smartphones used as *PSUs*.

Device	Processing capability	Cores (Family)	Wireless
SG_S4	Universal 5410 1.6 GHz GPU: PowerVR SGX544MP3	4(Cortax-A15)	Wi-Fi 802.11 a/b/g/n/ac, Dual-band, Wi-Fi Direct, GSM/HSPA, Bluetooth 4.0
GN_5	Qualcomm Snapdragon™ 800, 2.26 GHz, GPU: Adreno 330, 450 MHz	4 (Krait-400)	Dual-band, Wi-Fi Direct, GSM/CDMA/HSPA/LTE, Wi-Fi 802.11 a/b/g/n/ac, Bluetooth 4.0
DV_8	Intel® Atom™ CPU Z3480 2.13 GHz	2 (Bay Trail)	Wi-Fi 802.11 b/g/n, GSM/HSPA, Bluetooth 4.0

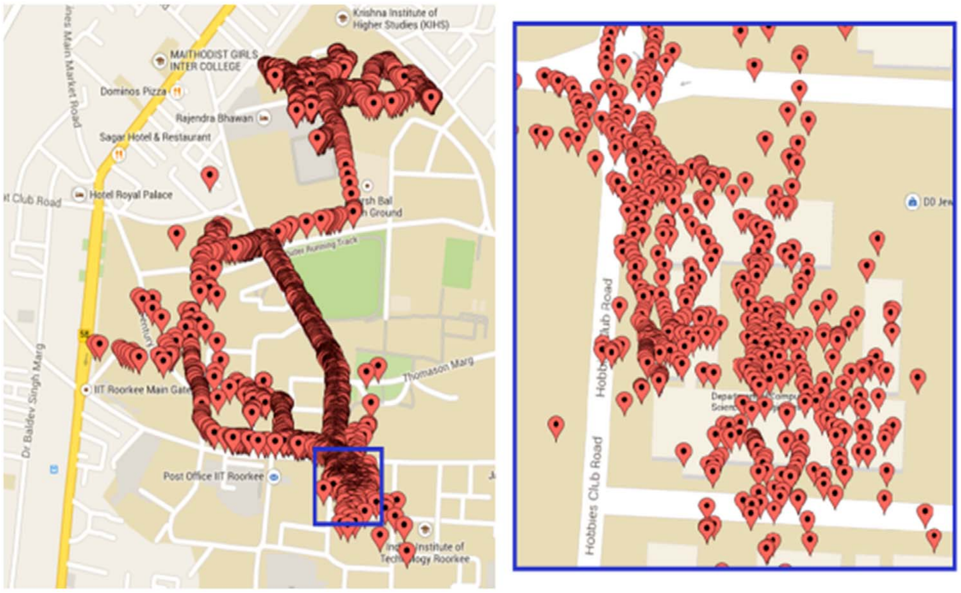


Fig. 9. Wi-Fi scanning to track people in IITR campus (Create a map, 2015).

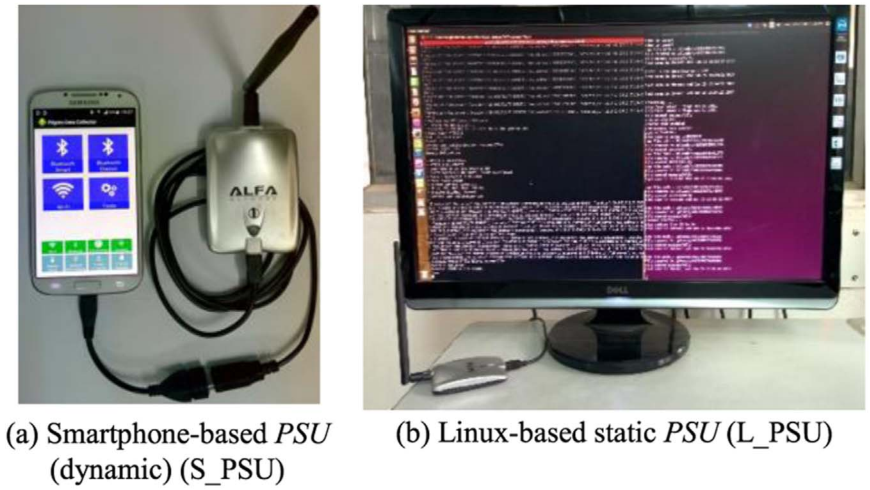

(a) Smartphone-based *PSU* (dynamic) (S_PSU)
(b) Linux-based static *PSU* (L_PSU)

Fig. 10. *PSUs* used for dynamic and static sensing.

and research institute in the state of Uttarakhand, India. It has a 1.48 km^2 campus housing many objects, such as academic departments, administrative buildings, student hostels, library, banks, post office, hospital, schools, canteen, cafeterias, shops, etc.

We have also collected data on 27 Feb., 2016 evening (*time* 18:00 to 20:00) at Har-ki-Paudi (Lat. & Long. - 29.9557771, 78.1711958), a religious place (river bank) in Haridwar, India where millions of pilgrims take the bath in the river Ganga on specific holy days, named *Exp@Haridwar*. We have tracked all those *C* devices using *SmartITS* for which the Wi-Fi is turned ON. We have three volunteers to collect data using *PSUs*. All *PSUs* are configured for channel 6, maximum records per file to be uploaded are 50, and duplicate detection rate is set to 5 min. *PSUs* keep moving most of the time. As a result, same MAC ids are rarely detected multiple times. In total, there are 7505 records collected and out of these 1877 MAC ids are unique.

7.3. Performance metric

We tested the *SmartITS* application against four performance metrics – *location accuracy, frame processing success ratio, battery consumption* and *response time*. Below, we define them formally.

- Location Accuracy (L_{acc}): It is a radius of the *PSU* where a user's presence can be estimated.

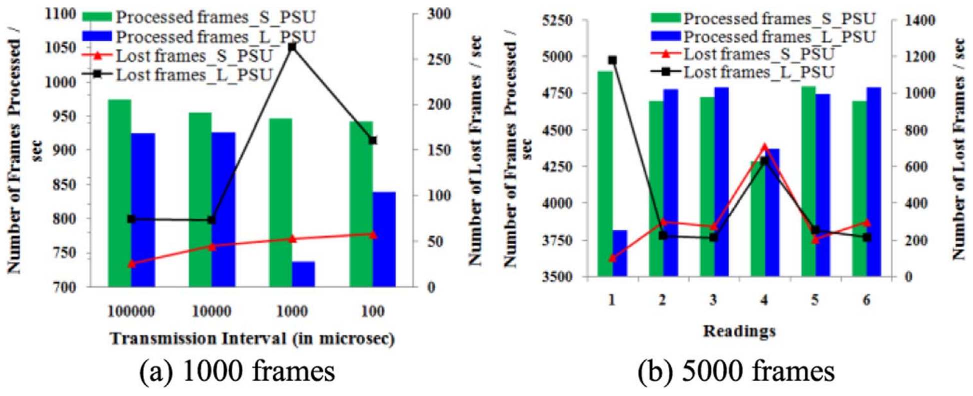


Fig. 11. *PSU's* processing capability with same MAC ids.

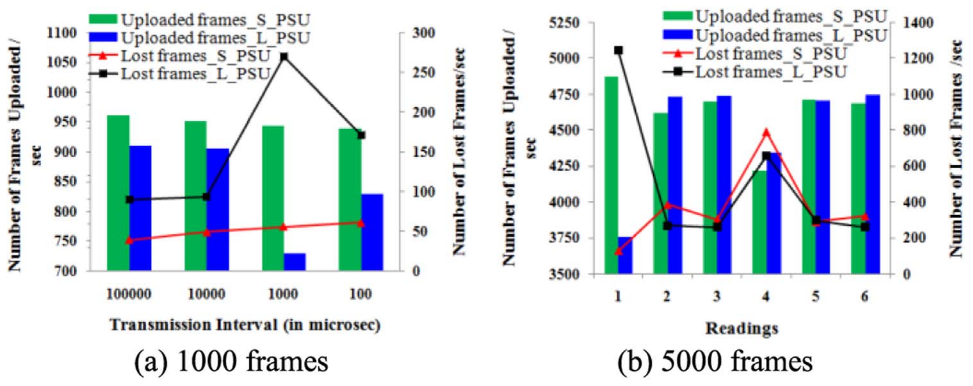


Fig. 12. *PSU's* uploading capability with unique MAC ids.

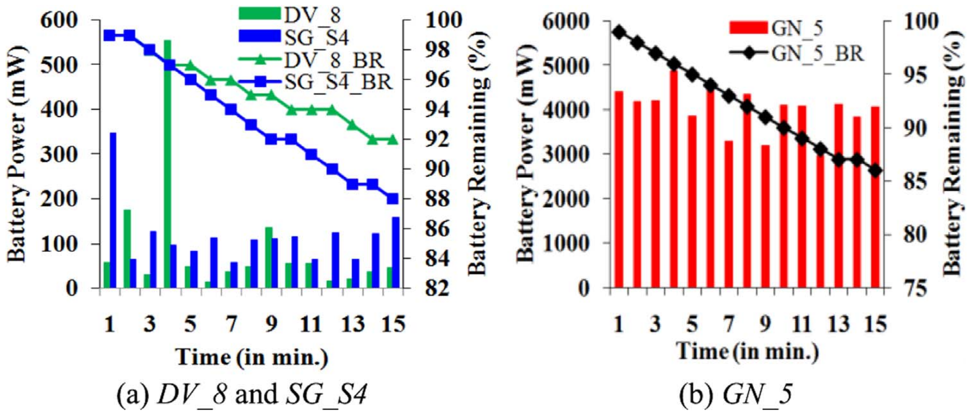


Fig. 13. Battery power consumption for processing and uploading of 1000 frames/s.

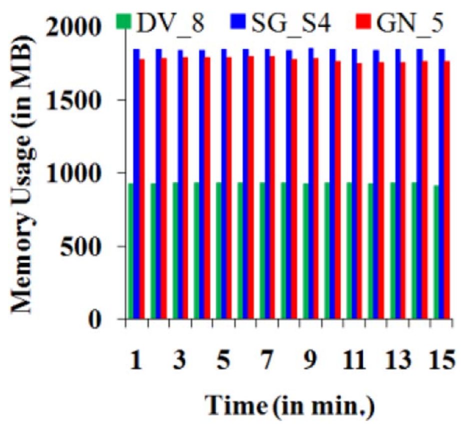


Fig. 14. Memory usage for processing and uploading of 1000 frames/s.

- Frame Processing Success Ratio (R_{succ}): It is the ratio between successfully processed frames (generated and sent by all *C*s covered) at a *PSU* to the total number of frames generated by all *C*s covered by the same *PSU*.
- Battery Consumption (E_B): It is the hourly rate of battery consumption by *C* (for generating probe requests) and *PSU* (for processing probe requests) devices.
- Response Time (R_t): It is the mean time elapsed between the instant a *C* sends a query to the C_{Server} and the instant *C* receives a response from the C_{Server} regarding his location.

R_t = *Round Trip Time* (RTT) + *Server processing time* (SPT), where *SPT* is the time required to process a request at the C_{Server}.

We have measured all, but the L_{acc} metrics is not computed for the simulation experiments as the Wi-Fi injector is a stationary computer. Similarly, for the testbed experiments, we could not calculate R_t as the C_{Server} is locally deployed and its IP is not accessible from outside world.

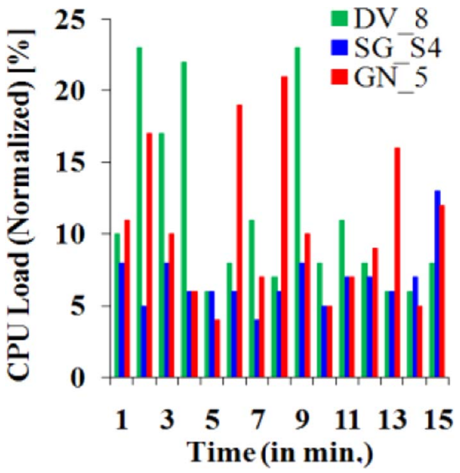


Fig. 15. CPU load (Normalized) [%] for processing and uploading of 1000 frames/s.

We further extend our analysis to find out the trade-off between battery consumption and location accuracy of GPS-based system, Cell-networks-based system and *SmartITS*. During the location tracking experiment, *SmartITS* does not perform any scanning (*Wi-Fi-BC-BLE*). We also calculate the battery power (energy) and performance analysis w.r.to CPU load, and memory usage of *SmartITS*, GPS-based system, and Cell-networks-based system in different scenarios.

For evaluating the performance of these systems, we use the power and performance profiling application, *Trepn^{TM} Profiler* ("Qualcomm, 2014b) developed by Qualcomm®. *Trepn* is 99% accurate (Hoque et al., 2016) which is highest among all existing performance profilers (Xu et al., 2013; Banerjee et al., 2014; Jung et al., 2012; "PowerTutor, 2017). *Trepn Profiler* collects 10 data samples per second (i.e., profiling interval 100 msec.). We consider four main performance metrics: *battery power* (for energy consumption), *memory usage*, *CPU load* (Normalized) and battery remaining percentage for performance and energy analysis of *SmartITS*, *GPS*, and *Cell-networks*-based systems in different scenarios. Below, we define these performance metrics formally:

- Battery Power (in mW) (Raw): It is the total battery power consumed by a specific Android-based application at the fixed sampling interval.
- Battery Remaining (in %): It is the loss in battery power (in %) for running the Android-based application on Smart device.
- Memory Usage (in MB): It is the total memory consumed by a specific Android-based application at the fixed sampling interval.
- CPU Load (Normalized): It is the CPU usage w.r.to the maximum potential of the CPU (i.e., CPU maximum frequency) for an Android-based application.

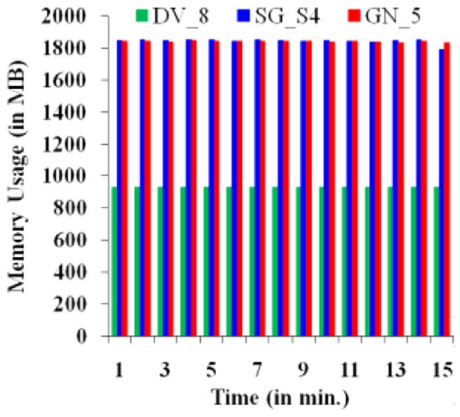


Fig. 17. Memory usage for processing and uploading of 5000 frames/s.

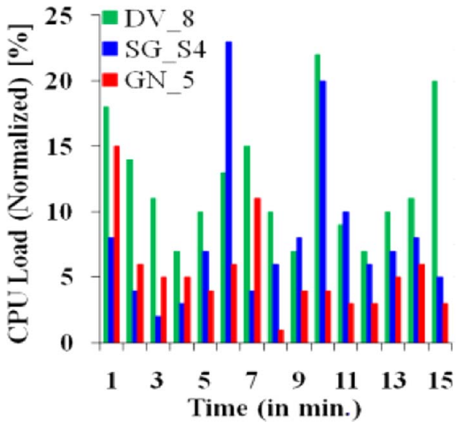


Fig. 18. CPU load (Normalized) [%] for processing and uploading of 5000 frames/s.

During the power and performance profiling, BT, GPS, Wi-Fi, and display screen state was always ON.

7.4. Simulation results

To evaluate the features and functionalities of *C* devices and *PSUs*, we categorize the experiments into *PSU side (Exp@PSU)* and *Client side (Exp@Client)* ones.

7.4.1. Exp@PSU

7.4.1.1. Processing capability of PSU: We performed two experiments to test the *PSU's* performance (using *SG_S4* as a *S_PSU* (see Fig. 10 (a))) when the *frame transmission rate* (FT$_{\text{rate}}$) by *C* devices is very high and the *PSU* is placed in a densely crowded area.

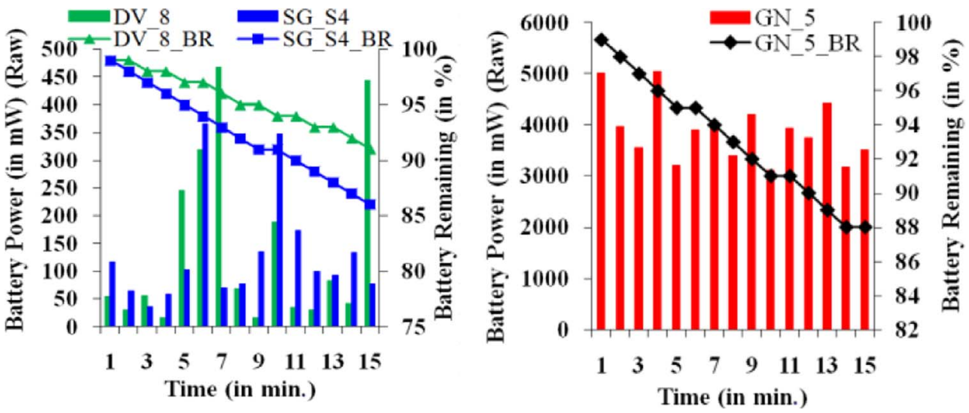


Fig. 16. Battery power consumption for processing and uploading of 5000 frames/s.

Table 4
Statistics for battery consumption w.r.to handling of 1000 no. of frames/s.

S. No.	Phone/ Tablet	Battery Capacity (in mAh)	Initial Charge (%)	Final Charge (in one h) (%)	Battery Consumption Rate/h (%)	Frames (Beacons) Processing Rate (%)	Frames (Beacons) Uploading Rate (%)
1	***SG_S4***	2600	99	55	44	97.4	96.1
2	***DV_8***	4100	99	71	28	96.5	94.9
3	***GN_5***	2300	99	47	52	97.1	96.2

Table 5
Statistics for battery consumption w.r.to handling of 5000 no. of frames/sec.

S. No.	Phone/ Tablet	Battery Capacity (in mAh)	Initial Charge (%)	Final Charge (in one h) (%)	Battery Consumption Rate/h (%)	Frames (Beacons) Processing Rate (%)	Frames (Beacons) Uploading Rate (%)
1	***SG_S4***	2600	99	47	52	97.96	97.48
2	***DV_8***	4100	99	67	32	95.28	93.87
3	***GN_5***	2300	99	43	56	96.9	96.12

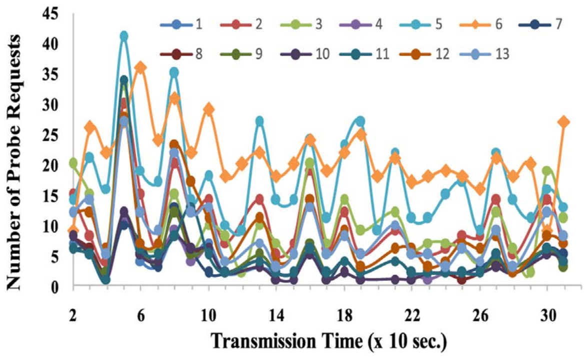


Fig. 19. Probe request transmission distribution on different channels [best viewed in colored image].

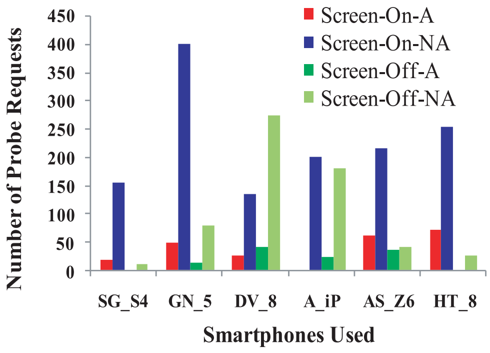


Fig. 21. Number of probe requests generated when mobile is A / NA with *AP*.

We also used a Linux-based *PSU* (L_PSU) for performing continuous logging of MAC ids (see Fig. 10 (b)).

To test whether a *PSU* can process frames fast enough, Wi-Fi injector transmits 1000 frames/s. with the **same** (representing same *C*) MAC ids at various transmission intervals ranging from 100 to 100,000 micro-seconds (see Fig. 11 (a)). Also, Wi-Fi injector transmits 5000 frames with same MAC ids at fixed transmission interval (see Fig. 11 (b)). On an average, frames captured are 954 out of 1000 and 4684 out of 5000 by *S_PSU* and 856 out of 1000 and 4547 out of 5000 by *L_PSU*. Therefore, the R_{succ} of *S_PSU* are 95.45% for 1000 frames and 93.68% for 5000 frames and R_{succ} of *L_PSU* are 85.68% for 1000 frames and 90.94% for 5000 frames.

Furthermore, to test the *PSUs* capability of handling the numbers of *C* devices present near the *PSU* (i.e., capability of *PSU* to upload incoming records to the C_{Server}), Wi-Fi injector transmits 1000 frames/sec with **unique** MAC ids (representing individual *Cs*) at various transmission intervals ranging from 100 to 100,000 micro-seconds (see Fig. 12 (a)) and 5000 frames/sec. having **unique** MAC ids (see Fig. 12 (b)). On an average, frames uploaded are 948 out of 1000 and 4632 out of 5000 by *S_PSU* and 843 out of 1000 and 4503 out of 5000 by *L_PSU*. Therefore, the R_{succ} of *S_PSU* are 94.8% for 1000 frames and 92.64% for 5000 frames and R_{succ} of *L_PSU* are 84.3% for 1000 frames and 90.06% for 5000 frames.

Figs. 13–15 show the results for battery power, battery remaining percentage (BR), memory usage and CPU load for processing and uploading 1000 frames/sec using all three *PSUs*. The average battery power for *DV_8*, *SG_S4* and *GN_5* are 154.995 mW, 190.159 mW and 3636.855 mW, respectively, where *DV_8* and *SG_S4* outperform *GN_5*. The average memory usage for *DV_8*, *SG_S4* and *GN_5* are 930.011 MB, 1847.018 MB and 1777.483 MB where *DV_8* is

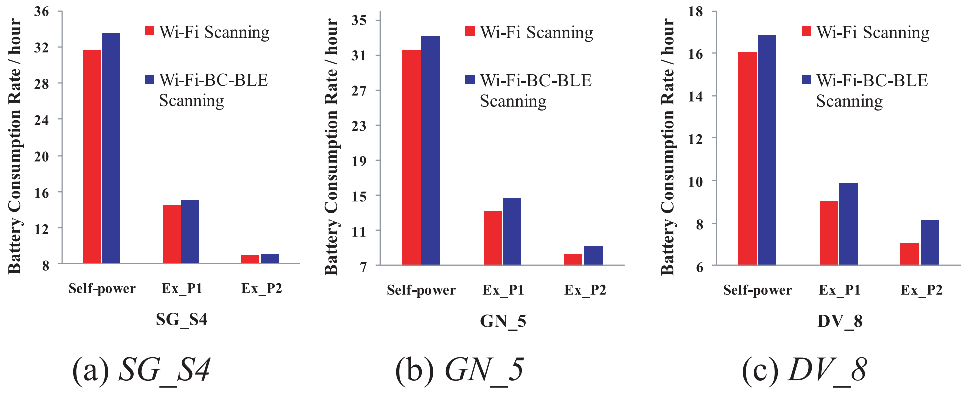


Fig. 20. Battery consumption rate at different Smartphones working as a *PSU*.

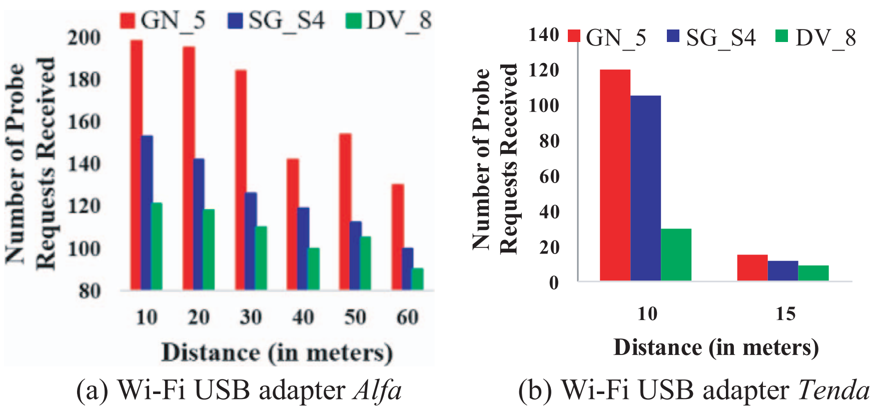


Fig. 22. Number of probe requests received at PSU w.r.to. distance.

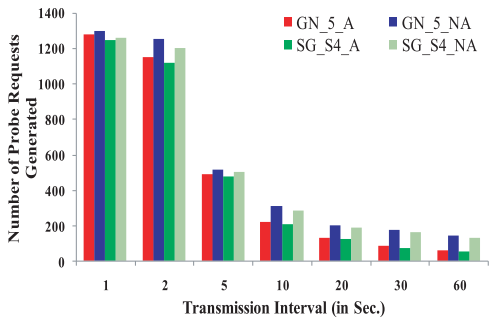


Fig. 23. Number of probe requests generation (for 10 min) w.r.to different transmission intervals.

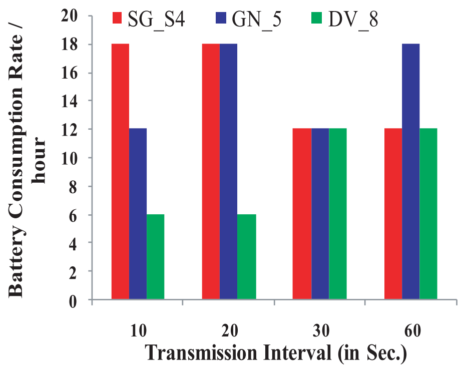


Fig. 24. Battery consumption rate per hour w.r.to different transmission intervals.

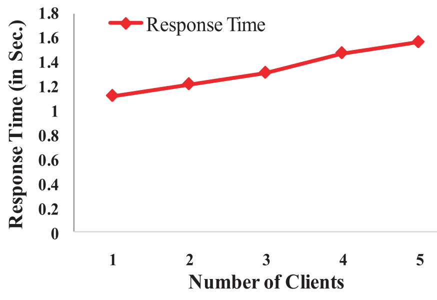


Fig. 25. Response time w.r.to number of *Cs*.

consuming significantly low memory in comparison to *SG_S4* and *GN_5*. The average CPU load for *DV_8*, *SG_S4* and *GN_5* are 15.428%, 9.796% and 8.240% where *DV_8* has high CPU load over *SG_S4* and *GN_5*. The results show that *DV_8* for *SmartITS* system (1000 frames/s) is performing well w.r.to battery power and memory usage, but performing poor w.r.to CPU load. *GN_5* performance is poor for both battery power and memory usage while *SG_S4* performance is not good w.r.to memory usage only.

Furthermore, to find the impact of increased number of frames on the *PSUs* performance, we consider another scenario in which processing and uploading rate is 5000 frames/s. Figs. 16–18 show the results for battery power, battery remaining percentage, memory usage and CPU load (Normalized) using all three *PSUs*. The average battery power for *DV_8*, *SG_S4* and *GN_5* are 165.291 mW, 208.241 mW and 3732.469 mW, respectively, where *DV_8* and *SG_S4* outperform *GN_5*. The average memory usage for *DV_8*, *SG_S4* and *GN_5* are 931.205 MB, 1847.064 MB and 1841.294 MB where *DV_8* is consuming significantly low memory in comparison to *SG_S4* and *GN_5*. The average CPU load for *DV_8*, *SG_S4* and *GN_5* are 16.307%, 8.640% and 6.626% where *DV_8* has high CPU load over *SG_S4* and *GN_5*. The results show that the performance of all three *PSUs* for handling 5000 frames/sec in comparison to 1000 frames/s is approximately same.

As it is not possible to find out immediate effect of processing 1000 and 5000 frames/s on battery consumption, we take a scenario in which *PSUs* battery are 99% charged and then they are used for continuous frames processing and uploading for an hour. Through this scenario, we found that *SG_S4*, *DV_8* and *GN_5* battery consumption rate/h for processing and uploading 1000 frames/sec is 44%, 28%, and 52%, respectively. While, the frame processing rate for *SG_S4*, *DV_8* and *GN_5* is 97.4%, 96.5% and 97.1%, respectively and frame uploading rate is 96.1%, 94.9% and 96.2%, respectively. For 5000 frames/sec processing and uploading, the battery consumption rate/hour for *SG_S4*, *DV_8* and *GN_5* is 52%, 32%, and 56%, respectively. While, the frame processing rate for *SG_S4*, *DV_8* and *GN_5* is 97.96%, 95.28% and 96.9%, respectively and frame uploading rate is 97.48%, 93.87% and 96.12%, respectively.

The results show that the battery consumption rate/hour is proportional to battery capacity of a Smartphone. On the other hand, when numbers of processing and uploading frames are increasing from 1000 frames/sec to 5000 frames/s, battery consumption rate/hour increased 23.8% for *SG_S4*, 14.3% for *DV_8* and 7.6% for *GN_5* which is well commensurate with handling high number of frames. Moreover, the frames processing and uploading rate for both 1000 and 5000 frames/s are same (approximately). Tables 4, and 5 show the statistics for battery consumption w.r.to handling of 1000 and 5000 no. of frames/s, respectively.

7.4.1.2. Wi-Fi channel selection: PSU detects the presence of a *C* through the probe requests transmitted by it. To find whether the probe requests are transmitted on all Wi-Fi channels or not, we attach 13 Wi-Fi adapters to a single Linux-based system using three 4-port USB hubs and one system's USB port. Each wireless interface was put in *monitor* mode and set to *listen on* a unique channel from 1 to 13.

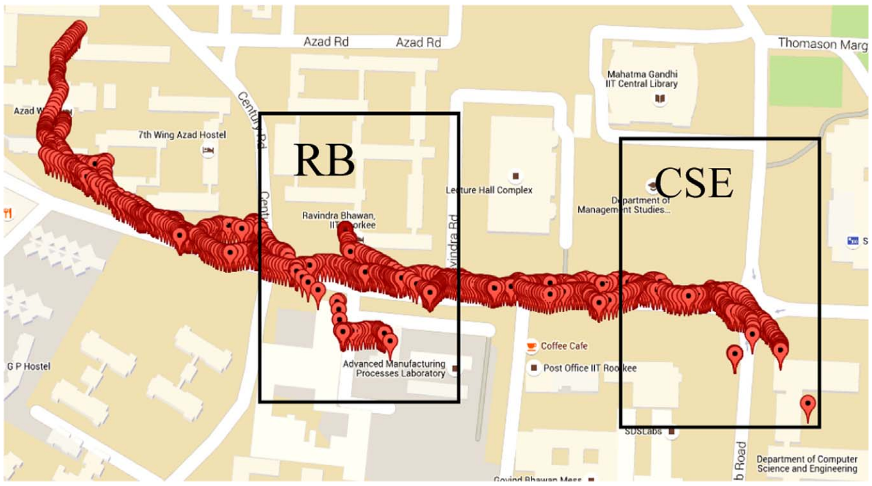

Azad Rd
Azad Rd
Thomason Marg
Mahatma Gandhi IIT Central Library
7th Wing Azad Hostel
RB
CSE
Lecture Hall Complex
Department of Management Studies
Ravindra Bhawan, IIT Roorkee
Coffee Cafe
Post Office IIT Roorkee
G P Hostel
Advanced Manufacturing Processes Laboratory
SDSLabs
Department of Computer Science and Engineering

Fig. 26. Location traces using GPS in indoor and outdoor environment at IITR campus.

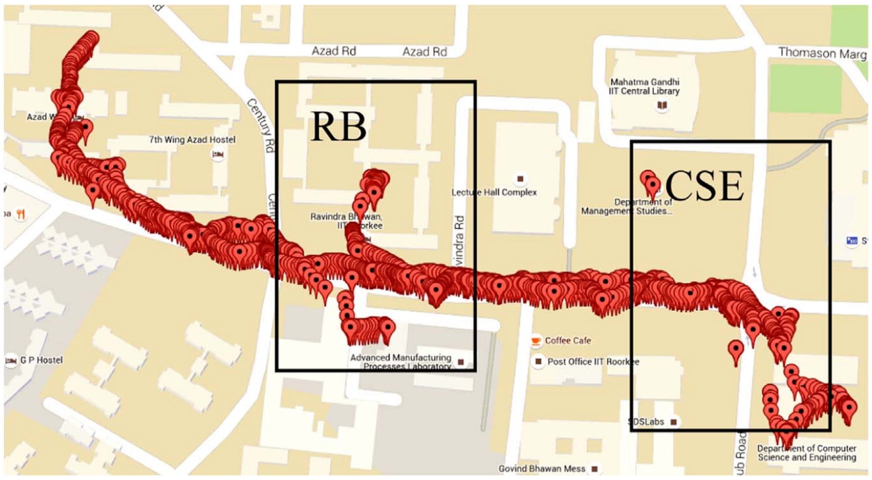

Azad Rd
Azad Rd
Thomason Marg
Mahatma Gandhi IIT Central Library
7th Wing Azad Hostel
RB
CSE
Lecture Hall Complex
Department of Management Studies
Ravindra Bhawan, IIT Roorkee
Coffee Cafe
Post Office IIT Roorkee
G P Hostel
Advanced Manufacturing Processes Laboratory
SDSLabs
Department of Computer Science and Engineering
Govind Bhawan Mess

Fig. 27. Location traces using *SmartITS* in indoor and outdoor environment at IITR campus.

All frames transmitted for a 5-min-duration on every channel in the 2.4 GHz band are captured using Tcpdump (TCPDUMP and LIBPCAP, 2015). Peaks in the Fig. 19 shows that the *C* periodically scans for *APs* and channel 5 & 6 register highest number of probe request transmissions. Therefore, *PSU* can adaptively switch to any channel (1–13) in order to increase the probability of detecting the probe requests from *C* devices. Furthermore, provision for multi-channel tracking helps to overcome country-specific limitations.

7.4.1.3. RTS injection: The current location of a *C* can be either found in C_{Server} or can be queried to all the *PSUs*. It can be possible that the C_{Server} does not have updated location information (as the *PSUs* are still operating inside logging period or the *C* device is operating in transmission interval for probes). Therefore, to get the latest location of a registered *C* during an emergency, C_{Server} can send a query to all deployed *PSUs*. *PSUs* ping nearby devices using *RTS* and get the *CTS* (Clear to Send) if that *C* is present there.

We used a Wi-Fi frame injector to send an *RTS* packet to the MAC id of the *C* being searched. We used *SG_S4* Smartphone and Linux-based system as two *C* devices. An experiment shows that *CTS* frames generated by *SG_S4* are much less compared to a Linux-based system. The reason is that most of the Smartphone uses the Wi-Fi in power saving mode. Another problem is that Smartphones can *listen* only on one channel at a time. Thus, tracking a *C* using *RTS* requires sending requests on all channels.

7.4.1.4. Battery consumption by PSU: Fig. 20 (a), (b) and (c) show the battery consumption (E_B) of different Smartphones working as *PSUs* during *plain Wi-Fi scanning* and *Wi-Fi-BT-BLE scanning*. We used *SG_S4*, *GN_5* and *DV_8* Smartphones under three different powering conditions - *self-powered*, *externally powered using Personal Computer's USB* (Ex_P1), and *externally powered using wall charger* (Ex_P2) for measuring E_B. The results show that *SG_S4* for *Ex_P2* during *Wi-Fi scanning* retains 38.18% and 71.63% battery power over *Ex_P1* and *self-powered*, respectively. *SG_S4* with *Ex_P2* during *Wi-Fi-BC-BLE scanning* retains 39.94% and 72.96% battery power over *Ex_P1* and *self-powered*, respectively. The same trend is visible for the other *PSUs* (refer to Fig. 20 (b) and (c)). The main aim for performing this analysis is to identify and to select the most suitable Smartphone(s) for *PSU* (for long-term scanning). We found through the experiments that *DV_8* outperforms *SG_S4* and *GN_5* w.r.to battery consumption rate.

7.4.2. Exp@Client

7.4.2.1. Smartphone's detection probability: To find the parameters required for increasing the detection probability of *Cs*, an experiment is conducted as shown in Fig. 21. The number of probes sent depends on the device OS, version, and the model. Therefore, to make the experiment more reliable, we have included three more Smartphones (*Cs*) which are *Apple iPhone 5* (A_iP), *Asus ZenFone 6* (AS_Z6) and

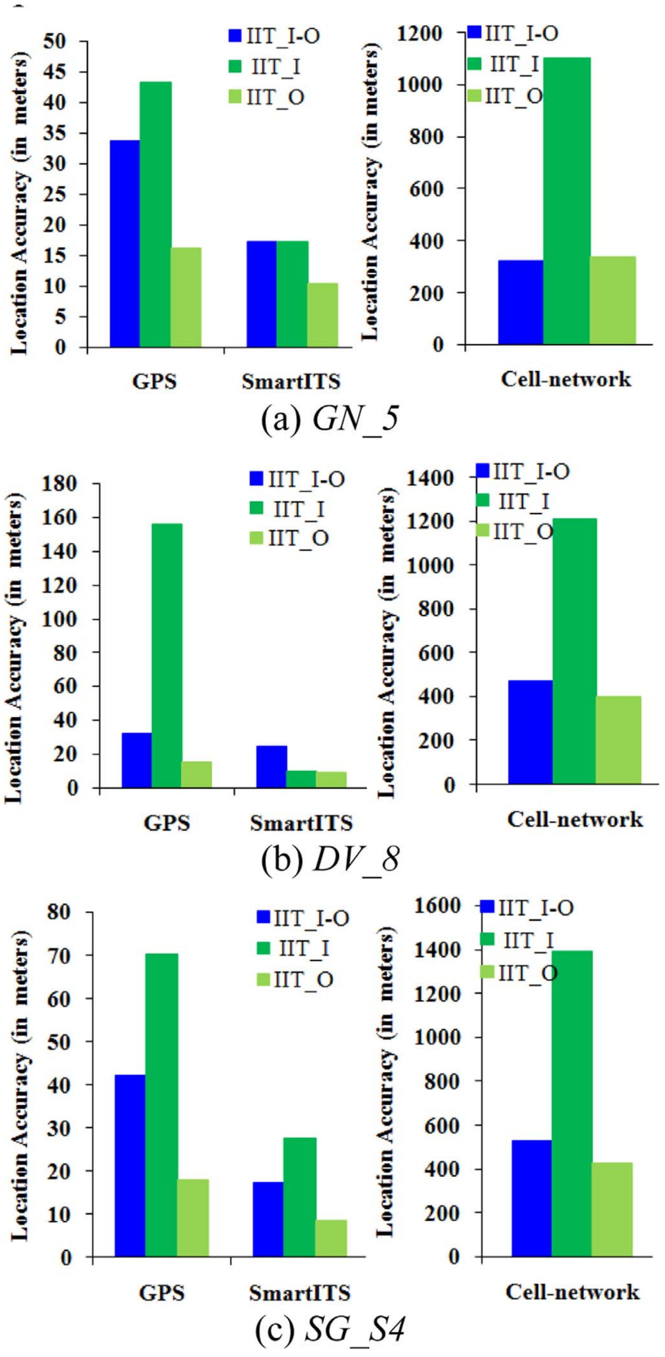


Fig. 28. L_{acc} for GPS, SmartITS and cell-network under various environments: *IIT_I*, *IIT_O*, and *IIT_I-O*.

HTC 8X (HT_8). Fig. 21 shows that maximum numbers of probe requests are detected when Smartphone's display is *on*, and the phone is *not associated* (NA) with any *AP*. It is also observed that transmissions of probe requests are low when Smartphone's screen is *locked* and it is *associated* (A) to an *AP*. While some of the Smartphones, such as *SG_S4* and *GN_5* transmit probe requests even their screens are *off*, and they are not connected to any *AP*. *SmartITS* makes the system energy-efficient by allowing *C* devices to transmit probe requests periodically even when the screen is *locked*.

We carried out an experiment for 10 min to show that the number of received probe requests degrade w.r.to increase the distance between *PSUs* and *C* devices (see Fig. 22 (a) and (b)). This decreases the probability of *C's* device detection over the distance. To handle this issue, *SmartITS* Client app provides a feature to adaptively increase the numbers of probe requests generation from *C's* device during an emergency (see Fig. 23). *SmartITS* Client app on *SG_S4* achieves around 41% improvement over an *SG_S4* without any such app.

Theoretically, the high number of probe requests can increase the battery consumption rate per hour. To check the impact of increased number of probe requests generation on the battery consumption, we calculate the battery consumption of using Client app for different Smartphones (*SG_S4*, *GN_5*, and *DV_8*) w.r.to increasing transmission intervals (see Fig. 24). It shows that installation of Client app has limited battery usage for all the Smartphones with increased number of probe requests generation.

7.4.2.2. *Response time:* To find the benefits of using REST API to upload and retrieve *C* data to/from C_{Server}, we calculated the R_t w.r.to number of *Cs* (n_Client). For one *C* (*SG_S4*), the response time of *SmartITS* is 1.122 s Fig. 25 shows the effect of *n_Client* on the response time. We have used three *SG_S4*, one *GN_5*, and one *DV_8*. The result indicates that the effect of *n_Client* is negligible on R_t where R_t for 2, 3, 4 and 5 *Cs* are 1.219 s, 1.313 s, 1.475 s and 1.566 s, respectively.

7.5. *Testbed experiment results*

To evaluate the features and functionalities of *C* devices and *PSUs* in real time, we categorize the experiments into two *Exp@IITR* and *Exp@Haridwar*.

7.5.1. *Exp@IITR*

7.5.1.1. *Location accuracy:* We compare the L_{acc} of *SmartITS* and GPS. Whenever a new *C* device is detected by a *PSU*, it logs the *C's* MAC id with its own location coordinates (latitude, longitude, and accuracy) obtained using the location API of Google Play Services and GPS-based Android App. If the location of a *PSU* is not available during *C's* MAC id detection, the *PSU* logs its last known location.

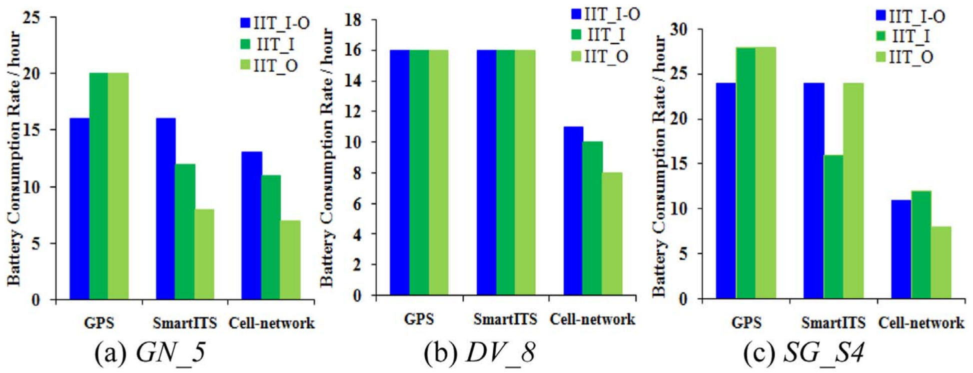


Fig. 29. Battery Consumption Rate/hour for GPS, *SmartITS* and Cell-network under various environments: *IIT_I*, *IIT_O*, and *IIT_I-O*.

Table 6
Statistics of *SmartITS* and GPS system for *IIT_I*.

System statistics	*IIT_I* (*SmartITS*)			*IIT_I* (GPS)		
	DV_8	*GN_5*	*SG_S4*	*DV_8*	*GN_5*	*SG_S4*
Average Battery Power [mW][Delta]	13.458	59.065	11.805	0.722	6.996	4.409
Average Location Accuracy (in m)	10.08	17.39	27.56	155.92	43.3	70.35
Average Memory Usage [MB]	929.313	1840.366	1847.019	932.331	1790.795	1848.743
Average CPU Load (Normalized) [%]	14.37467	5.583259	9.796903	11.12309	7.009994	6.9317

Table 7
Statistics of *SmartITS* and GPS system for *IIT_O*.

System statistics	*IIT_O* (*SmartITS*)			*IIT_O* (GPS)		
	DV_8	*GN_5*	*SG_S4*	*DV_8*	*GN_5*	*SG_S4*
Average Battery Power [mW][Delta]	20.810	27.251	24.317	16.401	15.104	12.774
Average Location Accuracy (in m)	9.49	10.47	8.46	15.79	16.25	17.96
Average Memory Usage [MB]	931.670	1833.525	1794.796	930.045	1776.914	1841.604
Average CPU Load (Normalized) [%]	22.93538	12.19	14.09749	13.08831	4.300628	8.406257

Table 8
Statistics of *SmartITS* and GPS system for *IIT_I-O*.

System statistics	*IIT_I-O* (*SmartITS*)			*IIT_I-O* (GPS)		
	DV_8	*GN_5*	*SG_S4*	*DV_8*	*GN_5*	*SG_S4*
Average Battery Power [mW][Delta]	6.568	42.239	8.854	7.811	10.246	7.930
Average Location Accuracy (in m)	24.69	17.33	17.30	32.81	33.83	42.18
Average Memory Usage [MB]	931.728	1832.933	1831.990	928.995	1774.807	1836.101
Average CPU Load (Normalized) [%]	14.952544	5.9733511	9.652374164	11.960266	5.2255889	8.292990809

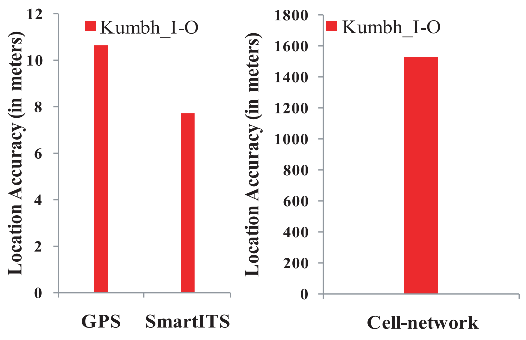


Fig. 30. L_{acc} for GPS, *SmartITS* and cellular-network of *Exp@Haridwar* (@indoor and outdoor).

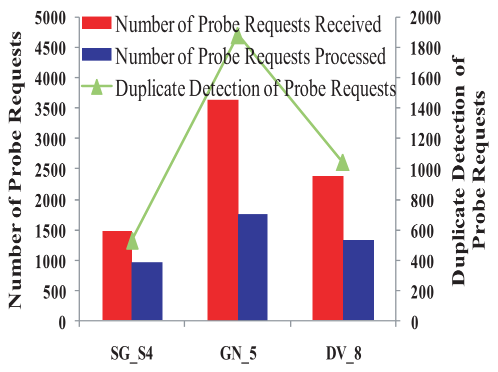


Fig. 31. Number of Probe requests processed and duplicate detection *Exp@Haridwar* (@indoor and outdoor).

In order to test the scalability and accuracy of *SmartITS* with respect to the processing capability of the *PSUs*, and location detection of the *C*, we defined a trajectory in the IITR campus comprising indoor and outdoor locations (Figs. 26 and 27). It can be observed that when a Plain GPS enabled Smartphone moves inside the Computer Science and Engineering (CSE) and Ravindra Bhawan (RB) buildings of IIT Roorkee (see Fig. 26), it (*SG_S4*) is able to log few location coordinates. This situation arises when GPS signals strength is weak. Fig. 27 shows that *SmartITS* can capture location coordinates inside both, *CSE* and *RB* buildings.

Aggregate location accuracy over a trajectory comprises of indoor and outdoor localization. The Aggregate value of L_{acc} using pure GPS for the entire trajectory is 14.413 m, while *SmartITS* is having 8 m of average L_{acc} through *S_PSU*. The accuracy improvement of *SmartITS* over GPS is 44.49%. Moreover, *Aggregate location accuracy* of a *PSU* using Cellular technology inside the IIT Roorkee campus is 528.47 m.

We extend the above-mentioned scenario in three categories to capture the L_{acc} in the IITR campus: *IITR-Indoor and Outdoor* (IIT_I-O), *IITR-Indoor* (IIT_I), and *IITR-Outdoor* (IIT_O) and further perform extensive experiments for these scenarios. *IIT_I* scenario means location logging and data processing is performed inside the campus buildings, such as Computer Science department, boys' hostels, etc. On the other hand, *IIT_O* scenario means identifying and tracking locations in open grounds, roads, and open corridor of buildings, etc. *IIT_I-O* is the combination of above two scenarios in which campus buildings, open grounds, roads, etc., are tracked randomly. *PSUs* are carried by mobile volunteers and all volunteers follow the same path at the same time for better analysis of experiments.

For the *IIT_I-O* environment, the aggregate values of L_{acc} using pure GPS are 32.81 m, 33.83 m, and 42.18 m for *DV_8*, *GN_5*, and *SG_S4*, respectively. Whereas, the aggregate values of L_{acc} using *SmartITS* are 24.69 m, 17.33 m, and 17.30 m for *DV_8*, *GN_5*, and

SG_S4, respectively. Therefore, the accuracy improvement of *SmartITS* over GPS is 24.75%, 48.77%, and 58.99% for *DV_8*, *GN_5*, and *SG_S4*, respectively (see Fig. 28 (a), (b), and (c)).

7.5.1.2. Battery consumption:. Fig. 29 shows the battery consumption rate per hour of *GN_5*, *DV_8*, and *SG_S4* in the *IIT_I*, *IIT_O*, and *IIT_I-O* scenarios for GPS, *SmartITS*, and Cellular-network. Battery

consumption of *SmartITS* is moderate in comparison to GPS while reasonably high w.r.to Cellular-network for all three PSUs/ Smartphones.

7.5.1.3. Power and performance analysis of SmartITS and GPS-based system: Tables 6–8 show the statistics of *SmartITS* and GPS system for *IIT_I*, *IIT_O*, and *IIT_I-O* environments w.r.to battery power, location accuracy, memory usage and CPU load performance metrics. Due to poor location accuracy, we did not consider Cell-network-based system for power and performance profiling. For better analysis of battery power, we consider battery power [Delta] instead of [Raw] where battery power [Delta] is preferred to calculate the effectiveness of improvement or impact of any change w.r.to baseline measurement. Through the results, we observe that battery power consumption is inversely proportional to location accuracy for all the *PSUs* and for all *IIT_I*, *IIT_O*, and *IIT_I-O* environments. Results show that *DV_8* outperforming *GN_5* and *SG_S4* in case of memory usage, while *DV_8* is having high CPU load in comparison to *GN_5* and *SG_S4*.

7.5.2. *Exp@Haridwar*

We performed extensive experiments using the real-time traces collected at Haridwar-Kumbh to validate the human tracking and identification using Smartphones, in both indoor and outdoor environments, named *Kumbh_I-O*.

7.5.2.1. Location accuracy (L_{acc}): The *Average* L_{acc} of a *PSU* computed using the GPS at *Kumbh_I-O* is 10.67 m while, *SmartITS* is providing 7.72 m of L_{acc}. The accuracy improvement of *SmartITS* over GPS is 27.65%. Moreover, average L_{acc} of a *PSU* using Cellular technology at *Kumbh_I-O* for the same track is 1531.26 m (see Fig. 30).

7.5.2.2. Processing capability of PSU: Fig. 31 shows the number of probe requests received and processed successfully at *PSU*. It also demonstrates the number of duplicate probe requests detected at the *PSU*. This result proves that the *PSU* is able to process the high number of received frames and can handle simultaneous connection of the large number of devices in real-time.

8. Conclusion and future works

To identify, monitor, and track the people in a large congregation seamlessly in both indoor and outdoor environments with high accuracy are the most challenging tasks of localization. Many existing techniques have many issues, such as signal multipath, additional hardware to deploy, Internet connection at Client side and significant accuracy error. In this paper, we proposed to develop a system, named *SmartITS*, for identifying, tracking and managing people in the large crowded place(s) using Wi-Fi through their Smartphones and BLE tags. The developed system does not require any Internet access on the Smartphones of the persons being tracked; neither does it require any application installation on the Client's Smartphone. As our system relies on idle Wi-Fi, it is power-efficient. Extensive experiments with a real-time testbed system in the large-scale congregation have shown that *SmartITS* is extremely scalable and can achieve up to 44.49% location accuracy than pure GPS-based systems for indoor-outdoor environment. Our experimental prototype illustrates the viability of our proposed system. MAC-ids and location tracking can be considered as the best possible solution for extracting, studying, and analyzing the human behavior based on their movements. In the future, we will try to extend *SmartITS* by analyzing human behavior in the large crowd, and to develop evacuation strategies during an emergency to make the cities smarter.

Acknowledgment

This work is partially supported by RailTel IITR Centre of Excellence in Telecom (RICET) at IIT Roorkee under the following grants: RCI-763(2)-ECD.

References

Al-Ali R., et al., 2008. Mobile RFID tracking system. In: Proceedings of the Information and Comm. Tech.: From Theory to Applications.

Alfa Wireless Adapter (AWUS036H), Wireless-G with SMA Adapter (2dBi or 5dBi Antenna), [Online]. Available: ⟨http://www.alfa.com.tw/products_show.php?Pc=34&ps=92⟩. (Accessed 25 November 2015).

Apache Web Server, [Online]. Available: ⟨www.apache.org/⟩. (Accessed 25 November 2015).

Banerjee A., Chong L.K., Chattopadhyay S., Roychoudhury A., 2014. Detecting energy bugs and hotspots in mobile apps. In: Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering, pp. 588–598.

Barbera, MarcoV.. et al., 2013. Signals from the crowd: uncovering social relationships through Smartphone probes. In: Proceedings of the Conference on Internet Measurement Conference. ACM.

Bonné, Bram, et al., 2013. WiFiPi: Involuntary Tracking of Visitors at Mass Events. IEEE WoWMoM.

Boukerche, A., et al., 2008. Secure localization algorithms for wireless sensor networks. IEEE Commun. Mag. 46 (4), 96–101.

Cheng, Ningning, et al., 2012. Inferring User Relationship From Hidden Information in Wlans. IEEE MILCOM.

Cisco Wireless Location Appliance, [Online]. Available: ⟨http://www.cisco.com/c/en/us/products/collateral/wireless/wireless-location-appliance/product_data_sheet0900aecd80293728.html⟩. (Accessed 31 January 2016).

Create a Map, [Online]. Available: ⟨http://www.mapcustomizer.com⟩. (Accessed 25 November 2015).

Cunche, Mathieu, et al., 2012. I Know Who You Will Meet This Evening! Linking Wireless Devices Using Wi-Fi Probe Requests. IEEE WoWMoM.

Deepesh, P.C.. et al., 2016. Experiences with using iBeacons for indoor positioning. In: Proceedings of the 9th ACM Conference on Software Engineering.

Di Luzio, et al., 2016. Mind Your Probes: De-Anonymization of Large Crowds through Smartphone WiFi Probe Requests. In Computer Communications, IEEE INFOCOM, 1–9.

Django Rest Framework, [Online]. Available: ⟨http://django-rest-framework.org⟩. (Accessed 25 November 2015).

Django Web Framework, [Online]. Available: ⟨https://www.djangoproject.com⟩. (Accessed 25 November 2015).

Faragher R.. et al., 2013. SmartSLAM–an efficient Smartphone indoor positioning system exploiting machine learning and opportunistic sensing. In Proceedings of ION GNSS.

Fazli S.. et al., 2011. Indoor positioning in Bluetooth networks using fingerprinting and lateration approach. In: Proceedings of the International Conference on Information Science and Applications.

Gallagher T.. et al., 2009. Trials of commercial Wi-Fi positioning systems for indoor and urban canyons. In: Proceedings of the symposium on GPS/GNSS (IGNSS).

Guido, D.A., et al., 2013. GNSS/cellular hybrid positioning system for mobile users in urban scenarios. IEEE Trans. Intell. Transp. Syst. 14 (1), 313–321.

HashMap, [Online]. Available: ⟨http://developer.android.com/reference/java/util/HashMap.html⟩. (Accessed 25 November 2015).

Hoque, M.A., Siekkinen, M., Khan, K.N., Xiao, Y., Tarkoma, S., 2016. Modeling, profiling, and debugging the energy consumption of mobile devices. ACM Comput. Surv. (CSUR) 48 (3), 39.

Huang J.. et al., 2011. Efficient, generalized indoor WiFi GRAPHSLAM. In: Proceeding of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1038–1043.

IEEE 802 LAN/MAN Standards Committee, 1999. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

Im, T., De, P., 2016. User-Assisted OCR on Outdoor Images for Approximate Positioning," In Information Science and Applications. ICISA Springer Singapore, 1419–1429.

Jung W., Kang C., Yoon C., Kim D., Cha H., 2012. DevScope: a nonintrusive and online power analysis tool for smartphone hardware components. In: Proceedings of the

Eighth IEEE/ACM/IFIP International Conference on Hardware/Software Codesign And System Synthesis, pp. 353–362.
Kulshrestha, T., et al., 2017a. An Improved Smartphone-Based Non-Participatory Crowd Monitoring System in Smart Environments. In: Proceedings of the IEEE International Conference on Computational Science and Engineering (CSE) and Embedded and Ubiquitous Computing (EUC), China, Vol. 2, pp. 132–139.
Kulshrestha T., et al., 2017b. A Fast and Scalable Crowd Sensing based Trajectory Tracking system. In: Proceedings of the IEEE International Conference on Contemporary Computing (IC3), India.
Libelium, Smartphone Detection, [Online]. Available: ⟨http://www.libelium.com/products/meshlium/Smartphone-detection/⟩ (Accessed 31 January 2016).
Lighttpd web server, [Online]. Available: ⟨https://www.lighttpd.net/⟩. (Accessed 25 November 2015).
Maghdid, H.S., et al., 2016. Seamless outdoors-indoors localization solutions on smartphones: implementation and challenges. ACM Comput. Surv. (CSUR) 48 (4).
Making Your App Location-Aware, Android Developers, [Online]. Available: ⟨https://developer.android.com/training/location/index.html⟩. (Accessed 20 November 2015).
Mantoro T.. et al., 2011. Hajjlocator: A hajj pilgrimage tracking framework in crowded ubiquitous environment. In: Proceedings of the IEEE International Conference on Multimedia Computing and Systems.
Mautz R., 2009. The challenges of indoor environments and specification on some alternative positioning systems. In: Proceedings of the WPNC, pp. 29–36.
Mitchell R.O.. et al., 2013. Hajj crowd management and navigation system: People tracking and location based services via integrated mobile and RFID systems. In: Proceeding of the IEEE ICCAT.
Mohandes M., 2011. Pilgrim tracking and identification using the mobile phone. In: Proceedings of the ISCE.
Musa, A.B.M., Eriksson, Jakob, 2012. Tracking unmodified Smartphones using wi-fi monitors. In: Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems.
Nginx web server, [Online]. Available: ⟨https://www.nginx.com/⟩. (Accessed 25 November 2015).
Ni, L.M., et al., 2011. RFID-based localization and tracking technologies. Wirel. Commun. 18 (2), 45–51.
Partyka J., A look at small indoor location competitors. GPS world. [Online]. Available: ⟨http://gpsworld.com/wirelesslook-small-indoor-location-competitors-13229/⟩ (Accessed 31 January 2016).
PowerTutor: A Power Monitor for Android-Based Mobile Platforms, [Online]. Available: ⟨http://ziyang.eecs.umich.edu/projects/powertutor/documentation.html⟩ (Accessed 25 May 2017).
Qualcomm, 2014b. Trepn Profiler. [Online]. Available: ⟨https://developer.qualcomm.com/trepn-profiler⟩ (Accessed 25 May 2017).
Raychoudhury, V., et al., 2015. Crowd-pan-360: crowdsourcing based context-aware panoramic map generation for Smartphone users. IEEE Trans. Parallel Distrib. Syst. 26 (8), 2208–2219.
Ren, L., et al., 2016. DE 2: localization based on the rotating RSS using a single beacon. Wirel. Netw. 22 (2), 703–721.
Rose, Ian, Welsh, Matt, 2010. Mapping the urban wireless landscape with Argos. In: Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems. ACM.
Sensewhere: Geo-fence technology and applications. Indoor Location Technology Leaders, [Online]. Available: ⟨http://www.sensewhere.com/images/geowhereDatasheet_compressed.pdf⟩ (Accessed 31 January 2016).
TCPDUMP and LIBPCAP, [Online]. Available: ⟨http://www.tcpdump.org/⟩. (Accessed 25 November 2015).
Tenda Wireless Adapter, [Online]. Available: ⟨http://www.tenda.cn/en/product/category-23.html⟩. (Accessed 25 September 2015).
Waadt A.. et al., 2009. An overview of positioning systems and technologies. In: Proceedings of the 2nd International Symposium on Applied Sciences in Biomedical and Communication Technologies,", pp. 1–5.
Waqar, et al., 2016. Smartphone positioning in sparse Wi-Fi environments. Comput. Commun. 73, 108–117.
WSGI module for Apache, [Online]. Available: ⟨https://github.com/GrahamDumpleton/mod_wsgi⟩. (Accessed 25 November 2015).
Xu, F., Liu, Y., Li, Q., Zhang, Y., 2013. V-edge: fast self-constructive power modeling of smartphones based on battery voltage dynamics. NSDI 13, 43–56.
Yang, C., et al., 2015. WiFi-based indoor positioning. IEEE Commun. Mag. 53 (3), 150–157.
Zandbergen, P., 2009. Accuracy of iPhone locations: a comparison of assisted GPS, WiFi and cellular positioning. Trans. GIS 13 (s1).

Tarun Kulshrestha is working towards the Ph.D. degree, Department of Computer Science and Engineering, at the Indian Institute of Technology (IIT), Roorkee, India. His research interests include Mobile Crowd Sensing and Computing, Internet of Things, Cloud Computing, Wireless Sensor Networks, and Mobile and Pervasive Computing. He is a member of IEEE.

Divya Saxena is working towards the Ph.D. degree, Department of Computer Science and Engineering, at the Indian Institute of Technology (IIT), Roorkee, India. Her research interests include Named Data Networking, Mobile Crowd Sensing and Computing, Mobile and Pervasive Computing, Networking, and Internet of Things. She is a member of ACM and IEEE.

Rajdeep Niyogi is an Associate Professor in the Department of Computer Science and Engineering, Indian Institute of Technology (IIT) Roorkee, India. He did PhD in Computer Science and Engineering from Indian Institute of Technology (IIT) Kharagpur, India, in 2004. His research interests include Automated Planning, Multiagent Systems, Distributed Systems, and Applications of logic and Automata theory. He is a member of ACM and ERCIM.

Vaskar Raychoudhury is an Assistant Professor in the Department of Computer Science and Engineering, Indian Institute of Technology (IIT) Roorkee, India. His research interests include mobile and pervasive computing and networking, Internet-of-Things, Wireless Sensor Networks. He received his PhD in Computing from The Hong Kong Polytechnic University in 2010. He is a senior member of ACM, and IEEE. He is a DAAD Fellow and Alexander von Humboldt Post-Doctoral Research Fellow.

Manoj Misra is a Professor and Head of Computer Science and Engineering Department at Indian Institute of Technology (IIT) Roorkee, India. He received his PhD from University of New Castle upon Tyne and has past experience of working as an Engineer at CMC Limited Noida, Assistant Engineer at Hindustan Aeronautic Limited at Kanpur India, Assistant Professor at HBTI, Kanpur, India. His research interests include Distributed Computing, Computer Networks, and Network Security and Cyber frauds.