Behaviour & Information Technology

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tbit20

On the exploitation of GPS-based data for real-time visualisation of pedestrian dynamics in open environments

Ahmed Alia, Mohammed Maree & Mohcine Chraibi

To cite this article: Ahmed Alia, Mohammed Maree & Mohcine Chraibi (2021): On the exploitation of GPS-based data for real-time visualisation of pedestrian dynamics in open environments, Behaviour & Information Technology, DOI: 10.1080/0144929X.2021.1896781

To link to this article: https://doi.org/10.1080/0144929X.2021.1896781

Published online: 11 Mar 2021.

Submit your article to this journal

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=tbit20

BEHAVIOUR & INFORMATION TECHNOLOGY
https://doi.org/10.1080/0144929X.2021.1896781

Taylor & Francis
Taylor & Francis Group

Check for updates

On the exploitation of GPS-based data for real-time visualisation of pedestrian dynamics in open environments

Ahmed Alia [a,b], Mohammed Maree [c] and Mohcine Chraibi [d]

[a]Institute for Advanced Simulation, Forschungszentrum Jülich, Jülich, Germany; [b]Faculty of Engineering and Information Technology, Computer Science Department, An-Najah National University, Nablus, Palestine; [c]Information Technology Department, Faculty of Engineering and Information Technology, Arab American University, Jenin, Palestine; [d]Institute for Advanced Simulation, Forschungszentrum Jülich, Jülich, Germany

ABSTRACT

Over the past few years, real-time visualisation of pedestrian dynamics has become more crucial to successfully organise and monitor open-crowded events. However, the process of collecting, efficiently handling and visualising a large volume of pedestrians' dynamic data in real time is challenging. This challenge becomes even more pronounced when pedestrians move in large-size, high-density, open and complex environments. In this article, we propose an efficient and accurate approach to acquire, process and visualise pedestrians' dynamic behaviour in real time. Our goal in this context is to produce GPS-based heat maps that assist event organisers as well as visitors in dynamically finding crowded spots using their smartphone devices. To validate our proposal, we have developed a prototype system for experimentally evaluating the quality of the proposed solution using real-world and simulation-based experimental datasets. The first phase of experiments was conducted in an open area with 37,000 square meters in Palestine. In the second phase, we have carried out a simulation for 5000 pedestrians to quantify the level of efficiency of the proposed system. We have utilised PHP scripting language to generate a larger-scale sample of randomly moving pedestrians across the same open area. A comparison with two well-known Web-based spatial data visualisation systems was conducted in the third phase. Findings indicate that the proposed approach can collect pedestrian's GPS-based trajectory information within 4 m horizontal accuracy in real time. The system demonstrated high efficiency in processing, storing, retrieving and visualising pedestrians' motion data (in the form of heat maps) in real time.

ARTICLE HISTORY
Received 22 May 2020
Accepted 19 February 2021

KEYWORDS
Real-time visualisation; pedestrian dynamics; crowd management system; GPS data; heat map visualisation

1. Introduction

With the rapid growth of population along with the increasing number of important events, dynamically finding less-crowded areas within large-scale events has become a major challenge for visitors as well as event organisers (Singh et al. 2020). As reported in Sharma et al. (2018), inefficient crowd management affects pedestrians' safety, their movement decisions and behaviours, and may also lead to increasing the efforts and costs required to efficiently monitor events with large numbers of pedestrians (Leopkey and Parent 2009). In order to plan successful events, organisers need to understand the visitors' movement characteristics and behaviour over time at an early stage (Zheng et al. 2014). Accordingly, greater attention has been given to study crowded events with huge numbers of pedestrians using a variety of tools and applications, such as pedestrians' devices that are used to collect their trajectories in an attempt to efficiently identify crowd density along events (Ebrahimpour et al. 2019). These approaches can be divided into three main categories (Ebrahimpour et al. 2019): crowd video analysis (Kulshrestha et al. 2019; Samsudin and Ghazali 2019), crowd social media analysis (Chaker, Al Aghbari, and Junejo 2017; Rizwan et al. 2018; Hu et al. 2019) and crowd spatio-temporal analysis (Gong et al. 2018; Xu, Li, and Fu 2019; Kong et al. 2020).

Developing systems under the first category plays an important role in analysing crowded event videos as they collect data about the movement features of pedestrians and assist in identifying abnormal behaviours during events (Sharma et al. 2018; Cheong et al. 2019). However, the large-size areas and high-density of pedestrians in such events make it difficult for such approaches to accurately respond in real time, leading to produce a large fraction of false positive results (Jabbari et al. 2019). In addition, the presence of obstacles, such as walls, trees and human bodies can block cameras from capturing and identifying correct abnormal

CONTACT Mohammed Maree mohammed.maree@aaup.edu
© 2021 Informa UK Limited, trading as Taylor & Francis Group

behaviours or even true pedestrian objects (Gowsikhaa, Abirami, and Baskaran 2014). Moreover, the cost and setup of cameras using this approach are normally high as reported in Ebrahimpour et al. (2019). On the other hand, approaches that fall under the second category use social media content, in addition to other important data, such as locations of user check-ins as their data source to study human movement behaviour (Ngo, Haghighi, and Burstein 2016). Using such data sources, these approaches generate spatio-temporal data points and location details that can be used to analyse crowds in large areas in cities, suburbs and urban areas (Tricco et al. 2018; Wu et al. 2018). However, it is important to point out that approaches of this category cannot make use of real-time collected data because the extraction of useful information from social media content is a very complex task (Stieglitz et al. 2018). In addition, social media data does not give information about human movement characteristics continuously at regular time intervals. For example, sharing check-ins allows users to mark and discuss places they visited (e.g. eating at local restaurants, shopping, visiting popular areas) as part of their social interaction online (Sharma et al. 2018; Rizwan et al. 2018). As such, we can conclude that approaches that fall under the first two categories may not generate sufficient information about human movement characteristics during high-density large-scale events in real time (Lamba and Nain 2017; Duives, Daamen, and Hoogendoorn 2018). Acknowledging these drawbacks, crowd spatio-temporal analysis approaches have proved to be an effective solution for collecting pedestrian data continuously during large-scale events (Zheng et al. 2014).

To efficiently achieve their goal, these approaches have utilised Global Positioning System (GPS) as their source of trajectory data acquisition (Zhao 2015). GPS indeed offers real-time response, easy and cheap to navigate feature, and has a good accuracy in outdoor areas (Konsolakis et al. 2018). All of these features contribute to making GPS an efficient option for real-time data collection about human movements in open areas and environments (Zhao 2015; Qureshi 2016). Recently, smartphones' built-in GPS receivers have been one of the most promising and convenient devices, especially, as they are widely used worldwide. The expected number of smartphones in 2020 is 2.5 billion (Helbostad et al. 2017). Also, smartphones can receive real-time data based on the current position of pedestrian's body at regular time intervals with good accuracy in outdoor areas (Konsolakis et al. 2018). As such, GPS and smartphones have allowed researchers to explore, investigate and monitor pedestrians' movement characteristics in outdoor areas more accurately and less invasively compared to other conventional approaches. In recent years, researchers have used mobile applications along with existing GPS data and web servers to develop real-time visualisation systems that can monitor pedestrian movement scenarios in open events (Wirz et al. 2012; Blanke et al. 2014). However, existing visualisation approaches suffer from a number of drawbacks and limitations as follows. First, with the increasing number of pedestrians in the same area, their performance degrades failing to cope with the real-time visualisation requirement. Second, the accuracy of the collected trajectories as well as visualisation tools is not tested under real-world conditions or using real-world scenarios. Third, using paid services or libraries increases system development costs. To address these limitations, we propose developing an efficient and low-cost system for online and real-time visualisation of pedestrian dynamics at open events/areas using GPS data. The system is aimed to be installed on users' smartphone devices to track pedestrian movements, collect and send GPS data to a web server in real time. Unlike existing approaches, the developed system is characterised by its efficiency in terms of the following aspects:

(1) The system collects pedestrians' trajectory data every second and estimates their horizontal accuracy to precisely depict the maps.
(2) New data processing with temporal storage of current pedestrian positions is used to improve the data visualisation performance.
(3) The system provides access to online real-time visualised maps that depict pedestrian movements along with real-time heat maps for spotting crowded areas.
(4) The system exploits efficient open source software to reduce the cost of its deployment.
(5) An archive of collected GPS data with multiple formats (SQL, JSON, and CSV) is provided for researchers interested in further developing and improving the current version of the proposed system. [1]

The rest of this paper is organised as follows. In Section 2, we explore and discuss a number of research works that are related to our proposed approach. Section 3 presents the proposed system and highlights its main components. In Section 4, we provide details on the experimental setup phase. Experimental results are then discussed in Section 5. Finally, Section 6 concludes

[1]The system's prototype and datasets can be accessed at https://github.com/PedestrianDynamics/.

this article and discusses the future extensions of our work.

2. Related work

Accurate real-time acquisition, processing, analysis and visualisation of pedestrian trajectory data during crowded events are the most crucial challenges for existing approaches, especially those that attempt to investigate and identify the most influential factors on pedestrians' dynamics and their movement behaviour and characteristics (Waga et al. 2012; Wirz et al. 2012; Waga et al. 2013; Blanke et al. 2014). Addressing these challenges helps event organisers make immediate decisions to avoid crowd accidents, and ensures proper monitoring and administration of event operations (Sharma et al. 2018). With the recent developments of smartphone industry, the possibility of tracking human positions via their GPS-enabled devices has been achieved, providing a good opportunity for developing low cost and real-time data processing and visualisation techniques that can assist in better understanding pedestrian dynamics at open areas that are characterised by their complex and irregular nature. Among the research works that were carried out in this domain is the work proposed by Waga et al. (2012) where the authors developed a system to track pedestrians using their smartphones. The collected tracking data was stored and visualised using google maps. In particular, they sent GPS tracking details to a web server every 30 seconds where they were stored using MySQL database management system. Then, to speed up the visualisation component, tracking data was updated into files every 24 h in worst case, depending on the users and the time range of tracks. Due to the time interval for updating files (which are the data source for the visualisation component), the system's visualisation component was inefficient in producing maps for the current collected GPS points at real time in the same manner as performed in Waga et al. (2013). In addition to that, Waga et al. (2012) and Waga et al. (2013) did not incorporate heat maps as part of the visualisation module, causing the visualising to be less effective and user friendly as we propose in our work.

Another visualisation technique was described in Blanke et al. (2014) to study the behaviour of crowds in a large festival (in 2013 at Zurich city in Switzerland) over a period of 3 days. A mobile application was developed in this context to collect users' locations continuously and visualise their presence during the festival. To do this, the researchers collected users' GPS data and sent it to a server every 2 s to be stored in a database. The stored data was processed at an interval of 2 min to be visualised in the form of maps. The authors experimentally tested the accuracy for several locations, and they obtained an accuracy between 150 and 500 m. According to the authors, the proposed approach was not developed for real-time pedestrian tracking and visualisation.

In Wirz et al. (2012), a festival's crowd conditions (crowd density, crowd turbulence, crowd velocity and crowd pressure) were visualised in real time using pedestrians' GPS location traces. To carry out this task, a mobile application was used to collect GPS locations and send them periodically after processing to a server which stores them in a database. In its heat map visualisation component, multiple mathematical models and methodologies were used to generate four different heat maps to infer four crowd conditions at specific time point. The main drawback of this approach is the computational complexity required to build the heat maps. In addition, the proposed system did not aim to detect the crowd conditions during the event, but only at specific time points.

In a similar line of research, Yun and Park (2015) analysed tourists' spatio-temporal behaviour at the rural festival in South Korea for 5 days. The authors developed a mobile application and a simple questionnaire to track the festival visitors and collect accurate spatio-temporal information about them. The questionnaire was conducted to know the visitors' socio-economic characteristics in addition to week day and weather to determine the effects of these characteristics on the crowd. In this context, each participant installed the mobile application to record his/her track and received a personally administered questionnaire. At the end of the experiments, participants reported back their data by submitting the questionnaires, in addition to uploading their tracking records (to a web server) in order for the researchers to depict and analyse the produced heat maps. However, the proposed approach did not collect and visualise the GPS tracks in real time, and the accuracy issue was not addressed by the researchers.

As we have highlighted in the above-mentioned discussion and to the best of our knowledge, the coupling of GPS data acquired using users' smartphones and heat maps for real-time visualisation purposes to understand the dynamics of pedestrians moving in large-scale open environments has been very little. Nevertheless, we can summarise the main drawbacks of existing approaches that have attempted to employ visualisation approaches and techniques for investigating the pedestrians' trajectory data as follows:

- Low accuracy of visualisation: The lack of accuracy measurements for collected positions has led to

lowering the accuracy of heat map visualisation. In general, smartphones are typically accurate within a 4.9-m radius under open sky and with normal conditions, however, their accuracy can degrade based on signal blockage, atmospheric conditions and GPS receiver quality (Specht et al. 2019; GPSUSA 2020). Therefore, it is necessary not to submit positions with low horizontal accuracy for heat map visualisation. Most of the existing approaches collected and provided the positions to the visualisation component regardless of their accuracy.

- High computational cost: Some of the existing approaches used all or a large portion of the collected data to visualise the crowd density of active pedestrians which resulted in increasing the required time to visualise the maps (Masiane et al. 2019).
- Inefficient storage and retrieval of GPS data points: Losing the newly collected GPS points in heat map visualisation degrades the quality of the produced maps. Some approaches attempted to address this drawback by selecting the required GPS points only for visualisation without searching in all data, but they did not apply this protocol automatically along with each process of data collection which led to losing current pedestrians' positions needed for heat map visualisation in real time.
- Real time: Some approaches did not process data in all their components in real time which led to an inefficient real-time approach.

In the next section, we introduce the architecture of our proposed approach and detail the methods and techniques that we employ to address the challenges of coupling GPS data points and heat map visualisation techniques.

3. Architecture of the proposed system

In this section, we first present the overall architecture of the proposed system. As depicted in Figure 1, data about each user's position is acquired every second and gets transferred to a dedicated web server for further processing, storage and visualisation. To do this, the system employs several components and modules as detailed below:

Mobile-based Data Acquisition: We have deployed the developed application on client smartphone devices for tracking their positions and sending this data to the web server for further processing. In particular, the application uses the smartphone's GPS sensor to determine users' positions while they are moving in an open space. After the user starts the application and agrees to use its associated GPS data collector for research purposes, the application starts collecting the current position of pedestrians moving at open events continuously (every second) and directly transfers the obtained GPS data over an active internet connection to the web server. The application remains active unless it is explicitly closed by the user. To ensure protecting users' privacy, the application does not collect any private information about the smartphone, as it just collects the positions (latitude and longitude) that are spotted inside the event's area to be visualised later in the form of heat maps. To further clarify this method, we present the algorithmic steps that we perform to carry out this task. As we can see in Algorithm 1, Lines 2–6 show how we acquire the GPS data which contains the latitude, longitude and horizontal accuracy.

Pre-processing of GPS Data: This component is employed for real-time processing of GPS data obtained from the mobile application as we described in the previous step. Algorithm 2 illustrates the methods that we utilise to carry out this step.

As we can see in Algorithm 2, the input is received in the form of the current user's position that is located within the event area. This data gets accordingly stored in two database tables (Archive and Live Tables, respectively) depending on different criteria as follows: in the Archive Table, processed user positions are always stored for data archival purposes. For all users' positions that are within a 'five meters' accuracy interval are dynamically transferred and stored in the Live Table after deleting the user's previous position from the table, more details about these tables are provided in the Data Repository Component.

Data Repository Component: This component is employed as a reference data repository for the collected positions of pedestrians inside the event. It namely consists of two relational database tables: (1) an Archive Table that works as a repository for all collected positions of moving pedestrians and (2) a Live Table which is a storage for the current positions of the pedestrians for real-time visualisation. In other words, the Live Table helps the proposed system to achieve real-time visualisation of the current users' positions as it contains a small number of positions which are equal to the number of current pedestrians. This means the computational time required for retrieving and processing GPS data from the Archive Table will be increasing over time while using the system. Therefore, using only one table for archiving and visualising is inefficient for real-time visualisation, because it retrieves all the previous/historical and current positions to return the current positions. The structure of the Archive and Live tables is presented in Table 1. Finally, this component can provide the GPS datasets in different formats

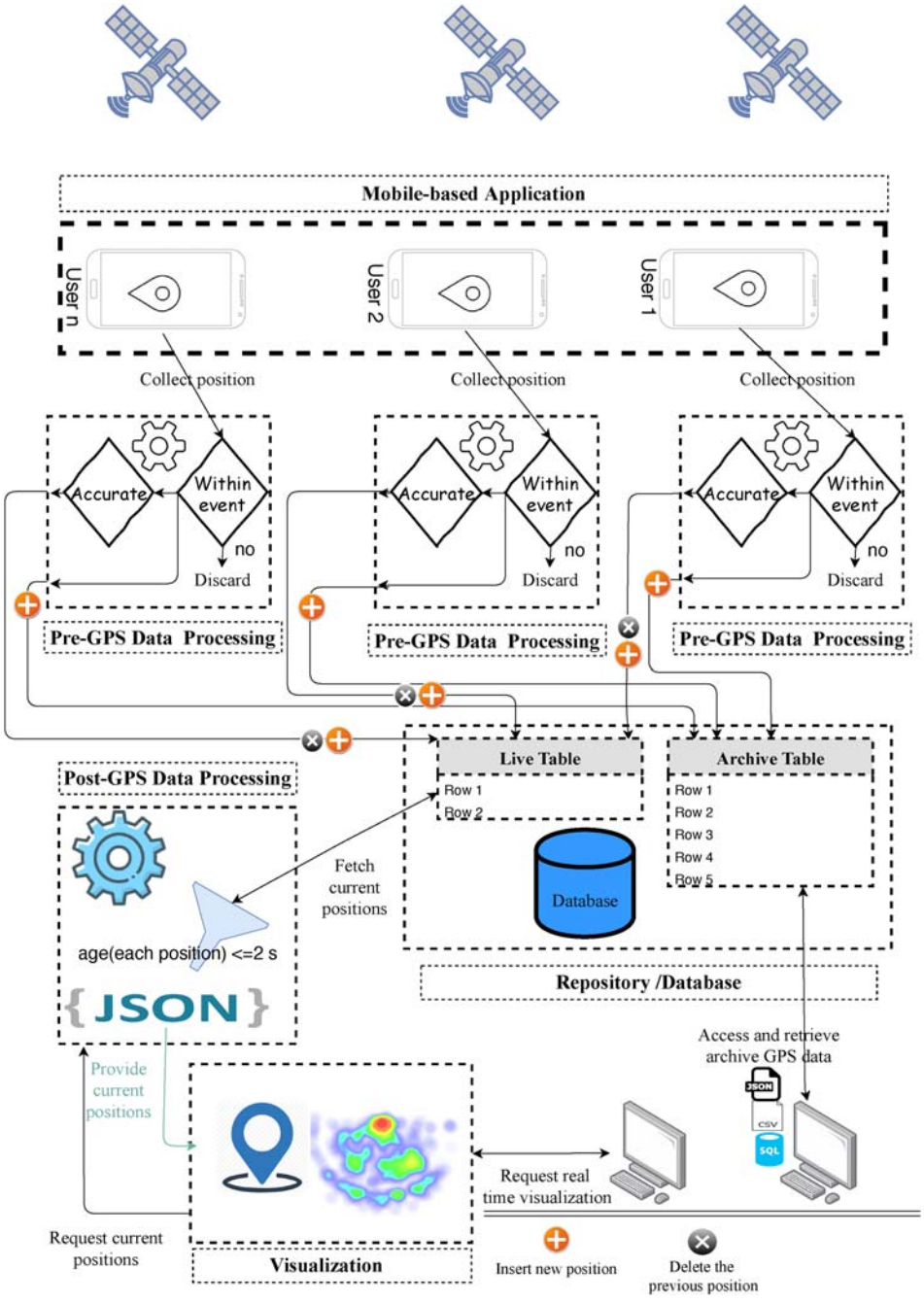

Figure 1. Overall architecture of the proposed system.

(SQL, CSV and JSON) to enable further processing of the produced data using a variety of tools and techniques.

Post-processing of GPS Data: This component assists the visualisation component to keep the Live Table up to date, retrieve and return its content in JSON format. As we discussed before, the Live Table aims to store the current positions of active pedestrians only during the event. Pre-processing of GPS data component updates the pedestrians' positions continuously, however, it cannot remove the last position of each pedestrian who has left the event from Live Table. As demonstrated in Algorithm 3 in lines 8–12, the post-processing of GPS data component removes the old positions before retrieving.

Real-time Data Visualisation: The main goal of this component is visualising pedestrians' trajectory data at real time asnormal/point maps and heat maps. It receives the current positions of all available pedestrians in the event from the post-processing component and visualises them on a web browser. Normal map is used to represent and update the current position of

Algorithm 1. Android-based mobile GPS data collection application.

Output:
record ← *array*(
['latitude']: latitude of the user's position,
['longitude']: longitude of the user's position,
['accuracy']: horizontal accuracy of the user's position,
['mobileId']: an identifier for the user's mobile)
▷ confirmation message for storing the record into the web server
conMsg ← NULL

Require: internet connection
1: **while** Application is running **do**
2: Location ← **Call** requestLocationUpdates(GPSprovider)
3: *record.latitude* ← Location.getLatitude
4: *record.longitude* ← Location.getLongitude
5: *record.accuracy* ← Location.getAccuracy
6: *record.mobileId* ← get aplication Id
7: **Print** record
8: create HttpPOSTConnection with the web server
9: open HttpPOSTConnection
10: submit record over HttpPOSTConnection
11: conMsg ← read HTTP response message
12: Prompt conMsg
13: **if** close button is clicked **then**
14: disconnect HttpPOSTConnection
15: break
16: **end if**
17: wait a second
18: **end while**

Algorithm 2. Pre-processing of GPS data.

Input:
//submitted record from mobile application
record ← *array*(
['latitude']: latitude of the user's position,
['longitude']: longitude of the user's position,
['accuracy']: horizontal accuracy of the user's position,
['mobileId']: an identifier for the user's mobile)
//Event area boundary
minLatitude ← 32.22677
maxLatitude ← 32.22955
minLongitude ← 35.21962
maxLongitude ← 35.22494
//Minimum accepted collected position accuracy for visualisation
minAccuracy ← 5
Output: msg ← NULL
1: **if** record.latitude ≥ minlatitude and record.latitude ≤ maxlatitude and record.longitudey ≥ minLongitude and record.longitude ≤ maxlongitude **then**
2: connect to database
3: ArchiveInsertSQL ← insert record into Archive Table
4: **if** execute(ArchiveInsertSQL) is successful **then**
5: msg ← record is Archived
6: **else**
7: msg ← record is not Archived
8: **end if**
9: **if** record.Accuracy≤minAccuracy **then**
10: **if** Live Table contains record from the same mobile **then**
11: execute(delete the record from the Live Table)
12: **end if**
13: liveInsertSQL ← insert record into Live Table
14: **if** execute(liveInsertSQL) **then**
15: msg ← msg + and is ready for visualisation
16: **end if**
17: **end if**
18: **Print** msg
19: disconnect from database
20: **end if**

Table 1. Data structure of archive and live tables.

Field	Description
id	A unique identifier for the position
mobileId	An identifier for the mobile
latitude	latitude of the user's position
longitude	longitude of the user's position
timestamp	Date and time for the obtained position
accuracy	horizontal accuracy of the user's position

each pedestrian as a point (longitude and latitude) on the map. In this context, points on the map will be updated automatically without updating the map in the background. On the other hand, we use heat maps (for visualising extensive point data sets) to continuously visualise and analyse large data sets and identify data clusters. This type of maps has demonstrated to be helpful in obtaining an overview of the current crowd density at a glance (Kuhfled 2017). In this context, a heat map is graphically depicted to represent spatial data where regions are coloured depending on measurement values found at the specific location (Wirz et al. 2012). In our work, the heat map represents hot and cold areas on the basis of pedestrians' densities in the same manner as performed by (Ihaddadene and Djeraba 2008). The hot areas (red) are regions where the density of pedestrians is high (more than 65%), the cold areas (blue) are regions where the pedestrians' density is low (less than 40%, see Mourner (2017) and Agafonkin (2019a)), and the yellow coloured areas are used to depict densities in between. Figure 2 shows an example of a normal map and a heat map that are obtained from real experiments that we have conducted at the open theatre at An-Najah National University. In Algorithm 3, lines 1-4, we detail the steps that are required to initialise the maps of the event, both normal and heat maps. In line 7, a request for the new positions from the Post-processing of GPS Data component is made toconvert them into

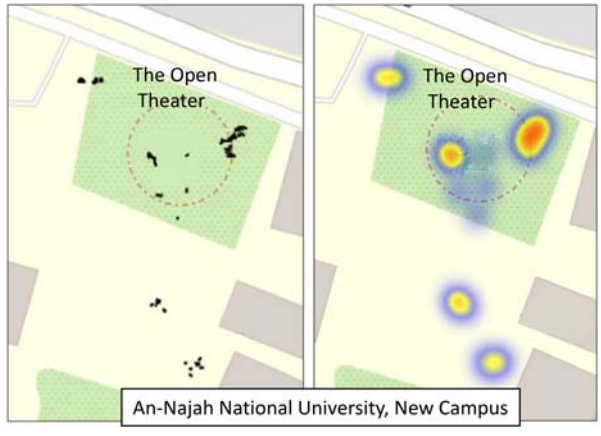

Figure 2. Screenshots: Left: normal map. Right: heat map (High density: red, low density: blue, between low and high: yellow).

JSON format (line 13). Then, both the normal map (lines 15–18) and the heat map (lines 20–21) are automatically updated.

4. Experimental setup

In this section, we present the setup and details of our experiments. In order to evaluate the performance of the proposed system, we have examined the following aspects:

(1) Efficiency of data collection module: Our goal in this context is to evaluate the system's ability to automatically collect pedestrians' trajectory data at every second, in addition to the accuracy of the collected GPS-based data in open areas.
(2) Real time data visualisation: Our aim here is to evaluate the run-time of the proposed system's prototyp by calculating the computational time for every component of the system, in addition to calculating the web page's loading time for the visualisation process for both normal and heat maps.

Algorithm 3. Post-processing and visualisation.

```
Input:
  //normal map or heat map
  mapType ← normal or heat
  //map updating interval time in second
  updatingDuration ← 1
  //close the visualisation

Output: normal map or heat map

 1: importing leaflet libraries (Agafonkin 2019b)
 2:    //Create an initial map of the center of the event
 3: map ← Call create map (Default Map center
    [32.2279565222561 36,35.22212731651962], Default Zoom: 18)
 4: Leaflet-providers← Call OpenStreetMap layer;
 5: previewing the initial map
 6: connect to database
 7: while request updates every second do
 8:    execute(delete records from Live Table where
       timeStamp=currentTime-2seconds)
 9:    data ← array()()
10:    while record←fetch record from Live Table do
11:       data.push(record.latitude,record.longitude)
12:    end while
13:    data←json.encode(data)
14:    reset map
15:    if mapType=normal then
16:       for i=0;i<length(data);i++ do
17:          marker(data[i]).addTo(map)
18:       end for
19:    else
20:       layer ← heatLayer(data)(Agafonkin 2019a)
21:       addLayerTo(map)
22:    end if
23:    if map is closed then
24:       break
25:    end if
26: end while
27: disconnect from database
```

We take an empirical approach to develop a prototype of the proposed system, conducting three types of experiments:

(1) A real-world experimental scenario where we used pedestrians' smartphone devices to collect their trajectory data in open areas (see Figure 3) and send them to a web server. The main goal of this experiment is to evaluate the efficiency of the data collection module and study the system's accuracy in collecting GPS-based trajectory data in open events/areas, as well as measuring the computational time of each component of the system's prototype. The experiment was conducted in several open areas at the new campus at An-Najah National University in Palestine. The focus was on the open theatre area. The area of the new campus is about 137.000 square meters, and its bounding box is identified by the following latitude and longitude: (32.22682, 35.22493), (32.2294, 35.2196). Both an open area with no high buildings and another open area that is surrounded by high buildings were selected. Figure 3 shows the open areas and the main routes of the tracked pedestrians. Nine users with different types of android-based smartphones have installed the application and participated in this experiment. They were divided into three groups. Each group started walking normally from different points as shown in Figure 3.
(2) A simulation experiment where we replaced pedestrians' smartphones with a PHP script to randomly generate a variable number of positions and submit them to the web server in an attempt to evaluate the server's computational time taken by the visualisation process for both normal and heat maps with more auto-generated trajectory data.
(3) A page loading time experiment where we compared the page loading time of the developed system's prototype with two well-known web-based spatial data visualisation systems in an attempt to evaluate the performance of the real-time visualisation component of our system.

We would like to point out that one of the main objectives of the proposed system is to perform efficiently in low cost. Therefore, the developed prototype only exploited efficient free/open source software in the same manner as reported in Bonaccorsi and Rossi (2003). In this context, we developed the android-based mobile application using Java and XML markup language. To implement the data processing components, we used PHP, JavaScript and JSON. In addition, we used

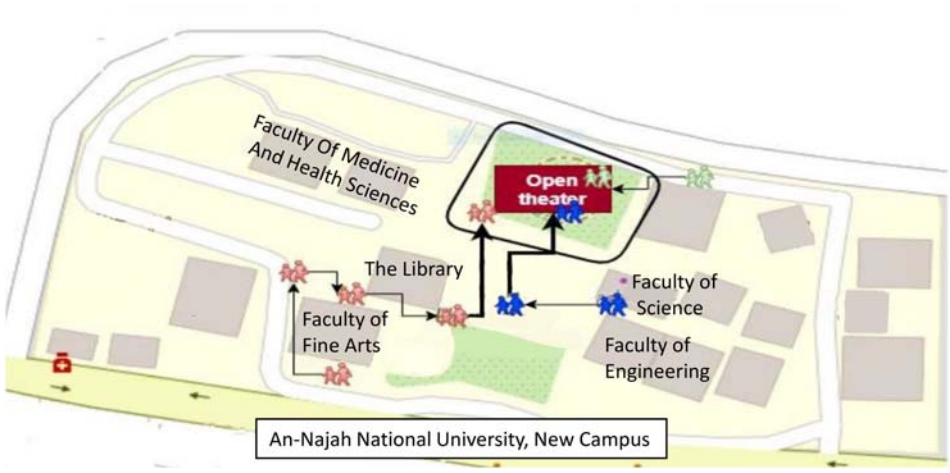

Figure 3. Real experiment area and the plan of pedestrians' movements.

MySQL as our database engine and PHP to implement the data repository component. For the data visualisation component, we employed Leaflet library (Agafonkin 2019b), which is one of the most well known and efficient open source JavaScript libraries for visualising normal and heat maps. In particular, the heat map used Leaflet.heat plugin (Agafonkin 2019a) which uses a simple-heat visualisation algorithm (Mourner 2017) that is combined with a point clustering technique to form a performance grid (Agafonkin 2019a). The simple-heat is a super-tiny JavaScript library for drawing heat maps with canvas focusing on simplicity and performance (Mourner 2017). Here, the grid is coloured with hot colour(red) when multiple points are close to each other and with cold colour (blue) otherwise. The current system's prototype is hosted on a server with one core, Intel(R) Xeon(R) Silver 4214 CPU 2.20GHz and 256 MB RAM. It can be accessed easily by any Android-based device with an internet connection. The android-based mobile application is installed on different types of android-based mobile phones. Figure 4 shows a screenshot of the android-based mobile application and a screenshot of the home page of the system's prototype. All client implementations were run on a personal computer running Windows 10 with (i5) 2.4 GHZ processor, and 8GB of memory.

5. Evaluation and results

5.1. Real-world scenario experimental results

This section aims to evaluate the efficiency of the data collection part by measuring the smartphone's GPS horizontal accuracy for collecting trajectory data positions at open event, and the required server computational time of each component in the proposed system's prototype.

5.1.1. Data collection part

The efficiency of the data collection part is discussed in this section, As shown in Table 2, 14,184 positions were collected in this experiment on 4 March 2020, the

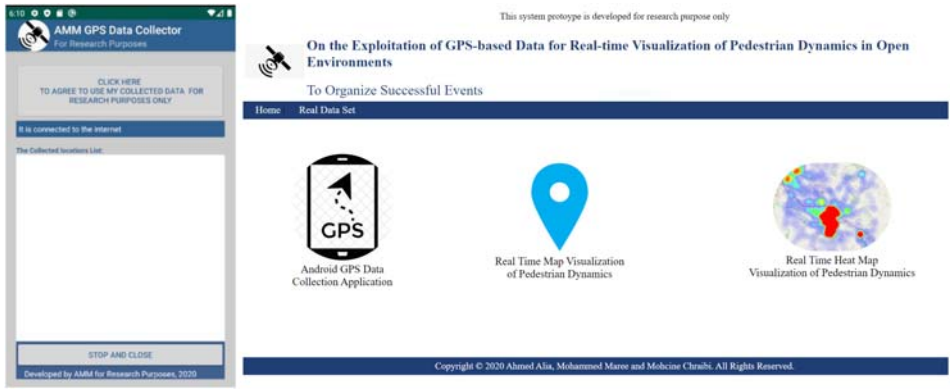

Figure 4. Screenshots: Left. Android-based mobile application. Right. Home page of the system's prototype.

experiment started at 11:27:40 and ended up at 11:56:54 based on local time of Palestine as illustrated in Figure 6. In Table 2, we also show a summary of the collected positions. The delay in connection to the internet and giving the permission to the android application by the user to start collecting the current position as well as the end permission caused a difference in the start and end times between the users, especially at the start time.

The duration column in Table 2 shows the experiment's duration in seconds for each user. It is an indicator for the expected number of collected positions, where each second means one collected position for each user. Users 2, 3, 5–9 achieved more than 99.5% of expected number of collected positions, while users 1 and 4 collected 94.32% and 97.53% of the positions, respectively. One reason for the failure in collecting one position at one second is the weak internet connection. Figure 5 presents comparisons between the collected positions and the expected number of collected positions for each user. Also, the average of collected positions is 98.8% at different circumstances such as: open area, open area surrounded by high buildings, different types of android-based smartphones and sometines poor internet connection. As a result, the android-based mobile application and the pre-processing of GPS data components were 98.8% efficient in collecting the current positions of pedestrians regularly at every one second.

Table 2. Summary of the collected positions.

User No.	Start time	End time	Duration (s)	No. of positions	Success %
1	9:27:40 AM	9:56:43 AM	1743	1644	94.32
2	9:28:03 AM	9:56:53 AM	1730	1724	99.65
3	9:28:12 AM	9:56:54 AM	1722	1722	100.00
4	9:29:47 AM	9:56:48 AM	1621	1581	97.53
5	9:31:44 AM	9:56:54 AM	1510	1510	100.00
6	9:30:19 AM	9:55:55 AM	1536	1534	99.87
7	9:28:13 AM	9:56:53 AM	1720	1695	98.55
8	9:29:32 AM	9:56:52 AM	1640	1632	99.51
9	9:37:47 AM	9:56:52 AM	1145	1142	99.74

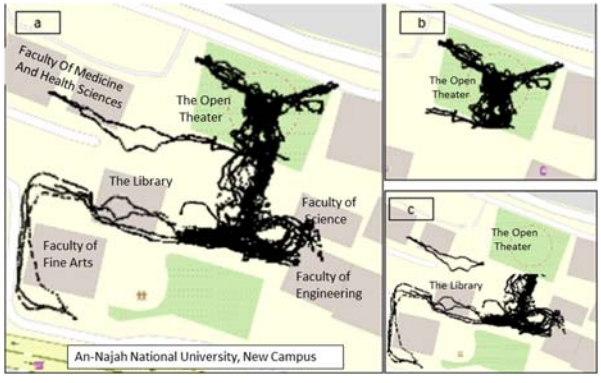

Figure 6. The real experiment area: (a) all area of experiment; (b) the open theatre area which does not contain high buildings; (c) this area is open with high buildings.

5.1.2. Accuracy evaluation

In this section, our goal is to investigate and measure the horizontal accuracy of the collected GPS data (latitude and longitude) by the android-based smartphones (i.e. smartphones' GPS horizontal accuracy).

In our prototype, we used the android location service to determine the current position with its estimated horizontal accuracy. In this context, the term horizontal accuracy is defined as the radius (in m) of 68% confidence (Kouřil and Šimeček 2020). In other words, if we draw a circle centred at this position's latitude and longitude, and with a radius equals to the accuracy value, then there is a 68% probability that the true location is inside the circle. Signal blockage, atmospheric conditions, and receiver design features/quality are the main local factors that affect GPS positioning accuracy (Padrón et al. 2017; GPSUSA 2020). Therefore, in the real experiment, we took these factors into consideration to simulate real-world scenarios. The area of the experiment is divided into two parts. The first area is the open theatre with no high buildings (Figure 6b), while the second area contains high buildings (Figure 6c). The goal of this division is to measure the influence of buildings on the accuracy of measurements using various android-based smartphone types, see Figure 6.

The results in Table 3 show the number of collected positions, expected average, the best and the worst horizontal accuracy measures for each user in both areas b and c. The total number of collected positions in the two areas is roughly the same (area b: 7000, area c: 7184). The average of the horizontal accuracy for all users in area b was within 4 m, while in area c, it was less than 5.18 m. These results show that the accuracy is affected by the presence of high buildings. To guarantee the accuracy of our system, the post-processing component only processes the accurately (within 5 m) collected positions for visualisation component. Figure 7

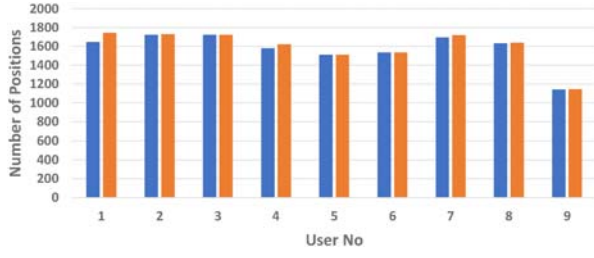

Figure 5. Comparison between the number of collected positions and the number of expected positions.

Table 3. Summary of the collected positions' accuracy.

	The Open Theatre (Area b)				Area c			
		Accuracy (m)				Accuracy (m)		
User No.	Positions	Average	Best	Worst	Positions	Average	Best	Worst
1	532	7.43	7.2	7.6	1112	7.48	9.7	7
2	931	2.87	1.5	7.5	793	4.03	12.5	1.5
3	922	3.78	3	9.5	800	4.49	24	2
4	900	4.92	2.5	9	681	6.01	14	2
5	558	2.47	1	3.4	952	2.52	4.1	1
6	1134	3.22	3.22	3.22	400	5.46	10.5	3.22
7	909	3.77	2	9	786	7.19	34.5	2.5
8	557	3.56	2	6	1075	4.24	17	2
9	557	6.86	5	9.5	585	8.15	20	5

shows the locations of the collected positions with an accuracy greater than 10 m. This low accuracy result is due to the presence of nearby high buildings.

Many of the open and large important crowded events are organised at open environment with very little signal blockage as the open theatre area in our study. Therefore, we are highlighting the results that were collected from the open theatre area. The significant difference in the average of accuracy between the users is depicted in Figure 8. As shown in this figure, the lowest accuracy value was recorded for User 1 with 7.43 m offset, while the highest accuracy value was marked for user 5 with 2.47 m. The accuracy measures for six other users (2, 3, 5, 6, 7, 8) were less than 4 m. We have considered the same area (area b) and the same weather conditions for all users, however, the quality of installed GPS sensors on the used smartphone devices was different, which explains the variance in accuracy measures.

Table 3 shows that the collected positions by users 1 and 9 are with more than 5 m accuracy. Meanwhile, other users collected positions with better accuracy. The main reason for that is the quality of the GPS sensor used by users 1 and 9 which is less efficient compared to others (Specht et al. 2019). As a result, modern android-based smartphones have a good ability to collect positions in open area with no high buildings within 4 m accuracy or less. In comparison with iPhone 6-based Avenza software for capturing horizontal positions at open area that is reported in Merry and Bettinger (2019), our system achieves better accuracy which is within 4 m, while the other system's accuracy is within 7–13 m.

5.1.3. Server computational time

This section aims to evaluate the server computational time of the proposed system's prototype based on the real experiment. In addition to accuracy aspects of the proposed system, in these experiments we have also considered another important aspect that plays a crucial role on the overall quality of the system. Our aim in this context is to evaluate the server's computational time of each of the various components of our proposed system's prototype. On one hand, the GPS data pre-processing component took 18.8 ms on average to pre-process each submitted position from the 14,184 positions and to store this into the two tables of the repository component (see Table 4). On the other hand, the computational time for post-processing the received GPS data and further visualising it in the forms of normal and heat maps for the nine users who were moving concurrently was recorded by the system. The total number of runs recorded in each component was 1130 runs. The average computational time for post-processing GPS

Figure 7. Locations of positions with accuracy greater than 10 m.

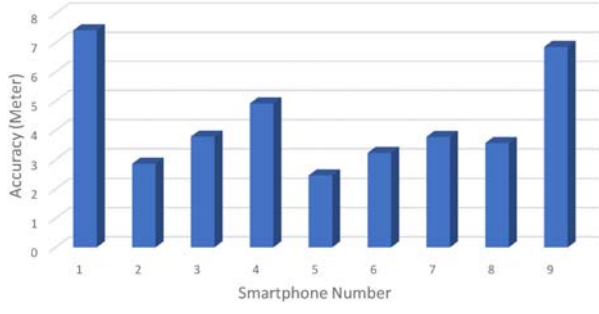

Figure 8. The average of accuracy for each user in the open theatre area (area b).

Table 4. The computational time of each component based on the real-world experiment.

Component	Number of runs	Computational time (ms)
Pre-Processing	14,184	18.793446
Post-Processing	1130	0.42608
Visualisation-Normal Map	1130	12.34
Visualisation-Heat Map	1130	8.8

data (to process and retrieve the current positions) and send them to the visualisation component for all runs was 0.43 ms. For the visualisation component, the average computational times needed to visualise normal maps and heat maps were 12.34 ms and 8.8 ms, respectively. The computational time difference between both techniques increases significantly when they are applied on more data points. See Figure 9. In Section 5.2, we demonstrate the impact of increasing the number of data points on the required time for visualising both types of maps.

The post-processing component took less time compared to the visualisation component for all runs. The main reason is because the pre-processing component (which is responsible for storing only the accurate positions within the specified area in a separate table) deletes the previous position of the same user from the data archiving table.

Based on the real-world experiment, the results show that the system's prototype can process, store and visualise the nine pedestrian positions in less than 35 ms. Moreover,, post-processing with normal and heat maps required about 41% and 34% of the computational time, respectively. In comparison, the pre-processing and the repository components required more than 50% of the computational time. However, these percentages will change when applied to larger datasets of trajectory points because every request in the pre-processing component is applied on one position (it processes multiple requests at the same time in parallel), while every request in the post-processing and visualisation components is applied on all current positions. This means that the total computational time of post-processing and visualisation components increases when the number of the current positions increases. Hence, the real-world experiment that was conducted to evaluate the system's prototype was not sufficient to evaluate the computational time of the post-processing and visualisation components. Therefore, in the next section, we test the post-processing and visualisation components over more positions.

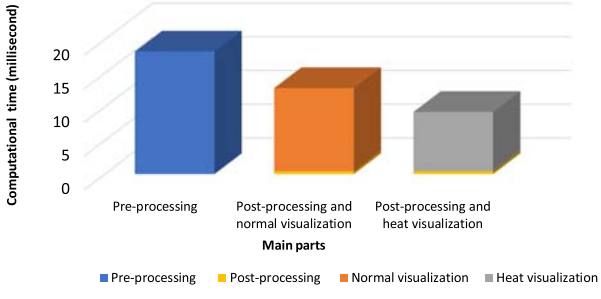

Figure 9. The computational time of the main components.

5.2. Simulation-based experimental results

In this section, we evaluate the server's computational time required by the post-processing and visualisation components over a greater number of positions. As we have pointed out in the previous section, the number of positions that can be handled by the post-processing and visualisation components affects the computational time of these components. In an attempt to demonstrate this effect, we have replaced the real-world android-based mobile application with a PHP script that generates a high number of positions randomly within the area of the open theatre and submitted them to a web server. Nineteen experiments that started from 100 positions and finished with 5000 positions were conducted to evaluate the computational time for the employed components over different data sizes. Table 5 shows the results of this step. To reduce the effect of fluctuations in the computational time, each experiment was repeated 20 times, and we took the average of all times for each run.

The results in Table 5 and Figure 10 show that the computational time for post-processing and normal and heat maps visualisation components increases when the number of positions increases. Figure 10 presents the computational time needed to visualise the current positions in heat map and normal map formats for 19 cases. Each visualisation process consists of two components: retrieving current positions (Live Table) by post-processing component and visualising them by visualisation component based on the map's format. The obtained results show that the post-processing component took very short computational time compared to the map visualisation component. Moreover, the computational time needed for visualising all positions in the largest case (5000 positions) in normal and heat maps were 1487 and 277 ms, respectively. The main reason of the difference in time between them are the leaflet plugins that are used by the system. Normal visualisation used marker plugin (Agafonkin 2019b) while heat map visualisation used heat map plugin (Agafonkin 2019a). The representation of each point with icon in marker plugin (Agafonkin 2019b) increases the computational time for loading and rendering, while

Table 5. Computational time of post-processing normal and heat maps.

Case	Positions num.	Computational time (ms)		
		Post-processing	Normal visualisation	Heat visualisation
1	100	0.99	136.55	120.2
2	200	1.4	158.2	119.2
3	400	1.956	211.56	122.7
4	600	2.698	256	125
5	800	3.262	311.86	146.2
6	1000	3.79106	356.8	155.2
7	1200	4.952	391	159.7
8	1400	4.99786	409	161.2
9	1600	5.8075	451	161.8
10	1800	6.74	504	161.9
11	2000	6.7908	553	163.6
12	2200	7.4439	571	163.4
13	2600	8.8449	655	175.4
14	3000	10.3908	822.63	184.8
15	3400	11.9399	883	205
16	3800	13.009082	1050	226.6
17	4200	13.8375	1140	234.2
18	4600	14.8353	1294	245.4
19	5000	16.479	1470	260.4

the heat map plugin does not use any icon for representation (Netek, Brus, and Tomecka 2019). In addition to that, Leaflet.heat plugin (Agafonkin 2019a) used a fast heat map visualisation algorithm (Mourner 2017). In Figure 10, normal map is affected significantly by increasing the amount of data. The time in normal map is amplified by a magnitute of 11 times, while it is doubled in heat map from the first case until the last case. Leaflet.heat plugin achieves much lower rendering times than the normal map (points map). According to Netek, Brus, and Tomecka (2019), the leaflet.heat plugin rendered a maximum of 3 million points in 16,313.7 ms, while normal maps (Agafonkin 2019b) rendered a maximum of 100,000 points. This means the heat map visualisation is more suitable for visualising a large amount of data in real time than normal maps. In addition to that, normal maps cover an entire map area, making each feature impossible to be identified (Netek, Brus, and Tomecka 2019).

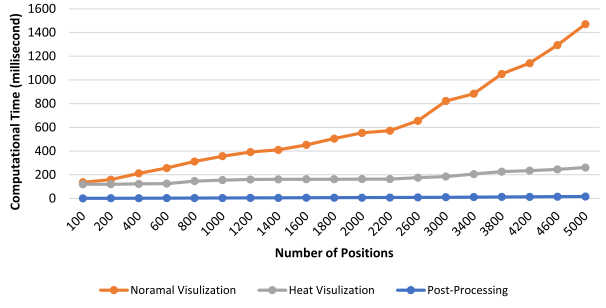

Figure 10. Computational time for post-processing and visualising normal and heat maps over different numbers of positions.

5.3. Page load time experimental results

In this section, we evaluate the performance of the real-time visualisation module of the proposed system's prototype. In particular, we compare the execution (a.k.a. Page Loading) time required by the proposed system's prototype, Maptive (M. team 2010–2020) and eSpatial (Espatial Team 2020) systems to load and display both normal and heat maps over 13 GPS data sets which are generated randomly. A page loading time in this context is defined as the average amount of time it takes for a page to show up on a client device screen, and it is calculated from the moment at which the user clicks on a page link or types in a Web address until the page is fully loaded on the client's browser (Shroff and Chaudhary 2017). This comparison criterion is highly variable due to different factors, such as client devices, network connection, in addition to other technical specification as reported in Shroff and Chaudhary (2017). Therefore, to make a fair comparison, we have used the same internet connection, client device, client browser and Chrome DevTools to measure the web page loading time. Both Maptive and eSpatial are efficient, available online and well-known real-time web-based spatial visualisation systems that support the visualisation of both normal and heat maps. To carry out the experiment, we used Chrome DevTools (C. D. team 2013–2020) to measure the web page loading time at each run. Each experiment was run 20 times, and we calculated the average results produced per each run.

Table 6, Figures 11 and 12 show the experimental results of web page loading time for the proposed system's prototype, Maptive and eSpatial web-based systems. As the results demonstrate, using the proposed system, we were able to achieve the best web page loading time required to visualise both normal and heat maps over all data sets. On average, the proposed system took 0.71% and 0.33% less than Maptive and eSpatial web-based system respectively, to visualise the normal map. Moreover, it needed 0.05% and 0.1% from the web page loading time that are needed in Maptive and eSpatial to visualise heat maps. In other words, our system's prototype visualised 5000 data points in heat map representation in less than a second while Maptive and eSpatial took 5.8 s and 9.2 s respectively. The main reason for these results is the efficiency of the Leaflet libraries for depicting the maps that are used in our system (Netek, Brus, and Tomecka 2019). In particular, we use a high-performance heat map visualisation algorithm combined with point clustering techniques to draw heat maps that are characterised by their simplicity (Mourner 2017). In other words, the efficiency of the

Table 6. Web page loading time required by our proposed system against Maptive and eSpatial systems.

		Maptive		eSpatial		The proposed system	
		Web page loading time (s)					
Data set	Positions num.	Normal map	Heat map	Normal map	Heat map	Normal map	Heat map
1	5000	9.204	9.708	4.968	5.856	4.458	0.847
2	4600	8.678	9.506	4.808	5.686	4.254	0.6962
3	4200	8.626	9.312	4.306	5.344	3.8	0.589
4	3800	8.59	9.224	3.712	5.262	3.2	0.5388
5	3400	8.412	9.132	3.538	5.164	3.018	0.4702
6	3000	8.114	9.07	3.468	4.824	2.516	0.3944
7	2600	7.994	8.878	3.186	4.156	2.344	0.3902
8	2200	7.84	8.592	3.154	3.938	1.856	0.387
9	1800	7.722	8.432	3.07	3.824	1.51	0.3666
10	1400	7.6	8.294	2.994	3.704	1.342	0.3544
11	1000	7.58	8.192	2.922	3.694	1.092	0.3438
12	600	7.566	8.166	2.828	3.618	0.79	0.3386
13	200	7.468	8.09	2.66	3.584	0.502	0.3382
Average		**8.11**	**8.82**	**3.51**	**4.51**	**2.36**	**0.47**

simple-heat algorithm makes the heat map visualisation fast (Mourner 2017).

6. Conclusion and future work

The wide usage of smartphones in Palestine and in the world motivates us to utilise smartphones to track and explore pedestrian dynamics at open events; to avoid potential crowd movement related incidents. In this work, we have proposed a real-time visualisation system for pedestrian dynamics at open events. The proposed system integrated smartphone's GPS sensor, web server and open source software to provide an efficient, online, accurate and low cost real-time system for collecting, storing and visualising the pedestrian movements. In addition, this paper presented the first smartphone-based GPS accuracy study in open events/areas in Palestine. To evaluate our approach, a prototype was developed and tested using a real-world experiment at different open areas, 19 simulation experiments, and 13 web page loading time experiments. The results demonstrated that our system collected the positions of pedestrians in open environment in real time and within an accuracy of 4 m. Moreover, it processed and stored the collected position in 18.8 ms, while retrieval and visualisation of heat maps for 5000 users took around 0.277 s. Also, the web page loading time in the developed system's prototype for heat map visualising for 5000 users was less than a second compared to Maptive and eSpatial which were 5.8 s and 9.2 s respectively.

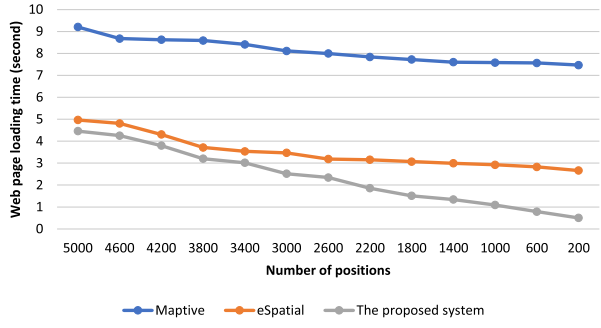

Figure 11. Web page loading time of normal map visualisation over different number of positions.

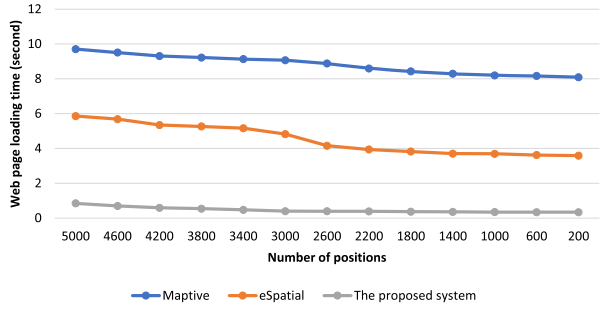

Figure 12. Web page loading time of heat map visualisation over different number of positions.

Based on the system's prototype implementation and initial results, our proposed system showed promising results. It is expected to track the pedestrian and vehicles in real time at open and large areas efficiently. Therefore, in the future work, we plan to develop an iOS-based mobile application to cover a larger-scale portion of pedestrians who use various mobile platforms, and conduct extensive real experiments using both android and iOS applications. We plan in this content to investigate the efficiency of our system on real large GPS data sets. In addition, we plan to develop a new neural network approach to predict pedestrians' movement behaviour and detect abnormal events at real time that may occur in large open events.

Acknowledgement

This work was supported by the German Federal Ministry of Education and Research (BMBF: Funding number

01DH16027) within the framework of the Palestinian-German Science Bridge project.

Disclosure statement

No potential conflict of interest was reported by the authors.

ORCID

Ahmed Alia http://orcid.org/0000-0002-3049-4924
Mohammed Maree http://orcid.org/0000-0002-6114-4687
Mohcine Chraibi http://orcid.org/0000-0002-0999-6807

References

Agafonkin, V. 2019a. "Leaflet.heat." https://github.com/Leaflet/Leaflet.heat (accessed on 30 June 2019).

Agafonkin, V. 2019b. "Leaflet: A Javascript Library for Interactive Maps." http://leafletjs.com/ (accessed on 5 February 2019).

Blanke, U., G. Tröster, T. Franke, and P. Lukowicz. 2014. "Capturing Crowd Dynamics at Large Scale Events Using Participatory GPS-Localization." In *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*, 1–7. IEEE.

Bonaccorsi, A., and C. Rossi. 2003. "Why Open Source Software Can Succeed." *Research Policy* 32 (7): 1243–1258.

C. D. team. 2013–2020. "Chrome Devtools".

Chaker, R., Z. Al Aghbari, and I. N. Junejo. 2017. "Social Network Model for Crowd Anomaly Detection and Localization." *Pattern Recognition* 61: 266–281.

Cheong, K. H., S. Poeschmann, J. W. Lai, J. M. Koh, U. R. Acharya, S. C. M. Yu, and K. J. W. Tang. 2019. "Practical Automated Video Analytics for Crowd Monitoring and Counting." *IEEE Access* 7: 183252–183261.

Duives, D., W. Daamen, and S. Hoogendoorn. 2018. "Monitoring the Number of Pedestrians in An Area: The Applicability of Counting Systems for Density State Estimation." *Journal of Advanced Transportation* 2018.

Ebrahimpour, Z., W. Wan, O. Cervantes, T. Luo, and H. Ullah. 2019. "Comparison of Main Approaches for Extracting Behavior Features From Crowd Flow Analysis." *ISPRS International Journal of Geo-Information* 8 (10): 440.

Espatial Team. 2020. "All-in-One Mapping Software".

Gong, Y., Z. Li, J. Zhang, W. Liu, Y. Zheng, and C. Kirsch. 2018. "Network-Wide Crowd Flow Prediction of Sydney Trains via Customized Online Non-Negative Matrix Factorization." In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 1243–1252.

Gowsikhaa, D., S. Abirami, and R. Baskaran. 2014. "Automated Human Behavior Analysis From Surveillance Videos: a Survey." *Artificial Intelligence Review* 42 (4): 747–765.

GPSUSA. 2020. "Official U.S. Government Information About the Global Positioning System (GPS) and Related Topics." https://www.gps.gov/systems/gps/performance/accuracy/ (accessed on 22 April 2020).

Helbostad, J. L., B. Vereijken, C. Becker, C. Todd, K. Taraldsen, M. Pijnappels, K. Aminian, and S. Mellone. 2017. "Mobile Health Applications to Promote Active and Healthy Ageing." *Sensors* 17 (3): 622.

Hu, Q., G. Bai, S. Wang, and M. Ai. 2019. "Extraction and Monitoring Approach of Dynamic Urban Commercial Area Using Check-in Data From Weibo." *Sustainable Cities and Society* 45: 508–521.

Ihaddadene, N., and C. Djeraba. 2008. "Real-Time Crowd Motion Analysis." In *2008 19th International Conference on Pattern Recognition*, 1–4. IEEE.

Jabbari, A., K. J. Almalki, B.-Y. Choi, and S. Song. 2019. "Icemocha: Intelligent Crowd Engineering Using Mobility Characterization and Analytics." *Sensors* 19 (5): 1025.

Kong, X., H. Gao, O. Alfarraj, Q. Ni, C. Zheng, and G. Shen. 2020. "Huad: Hierarchical Urban Anomaly Detection Based on Spatio-temporal Data." *IEEE Access* 8: 26573–26582.

Konsolakis, K., H. Hermens, C. Villalonga, M. Vollenbroek-Hutten, and O. Banos. 2018. "Human Behaviour Analysis Through Smartphones," In *Multidisciplinary Digital Publishing Institute Proceedings*, vol. 2, 1243.

Kouřil, P., and M. Šimeček. 2020. "Usability of Wi-fi Fingerprint Approach for Place Departure Recognition in Travel Surveys." *Travel Behaviour and Society* 18: 83–93.

Kuhfled, W. 2017. *Heat Maps: Graphically Displaying Big Data and Small Tables*. Cary, NC: SAS Institute Inc.

Kulshrestha, T., D. Saxena, R. Niyogi, and J. Cao. 2019. "Real-time Crowd Monitoring Using Seamless Indoor-outdoor Localization." *IEEE Transactions on Mobile Computing* 19 (3): 664–679.

Lamba, S., and N. Nain. 2017. "Crowd Monitoring and Classification: A Survey." In *Advances in Computer and Computational Sciences*, 21–31, Springer.

Leopkey, B., and M. M. Parent. 2009. "Risk Management Issues in Large-scale Sporting Events: A Stakeholder Perspective." *European Sport Management Quarterly* 9 (2): 187–208.

Masiane, M. M., A. Driscoll, W. Feng, J. Wenskovitch, and C. North. 2019. "Towards Insight-driven Sampling for Big Data Visualisation." *Behaviour & Information Technology* 39 (7): 788–807.

Merry, K., and P. Bettinger. 2019. "Smartphone Gps Accuracy Study in An Urban Environment." *PloS One* 14 (7): e0219890.

Mourner. 2017. "Leaflet.heat." https://github.com/mourner/simpleheat (accessed on 24 Jul 2017).

M. team. 2010–2020. "Web-Based Mapping Software Platform".

Netek, R., J. Brus, and O. Tomecka. 2019. "Performance Testing on Marker Clustering and Heatmap Visualization Techniques: A Comparative Study on Javascript Mapping Libraries." *ISPRS International Journal of Geo-Information* 8 (8): 348.

Ngo, M. Q., P. D. Haghighi, and F. Burstein. 2016. "A Crowd Monitoring Framework Using Emotion Analysis of Social Media for Emergency Management in Mass Gatherings." *arXiv preprint arXiv:1606.00751*.

Padrón, G., T. Cristóbal, F. Alayón, A. Quesada-Arencibia, and C. R. García. 2017. "System Proposal for Mass Transit Service Quality Control Based on Gps Data." *Sensors* 17 (6): 1412.

Qureshi, S. 2016. "Creating a Better World With Information and Communication Technologies: Health Equity".

Rizwan, M., W. Wan, O. Cervantes, and L. Gwiazdzinski. 2018. "Using Location-based Social Media Data to Observe Check-in Behavior and Gender Difference: Bringing Weibo Data Into Play." *ISPRS International Journal of Geo-Information* 7 (5): 196.

Samsudin, W. N. A. W., and K. H. Ghazali. 2019. "Crowd Behavior Monitoring Using Self-adaptive Social Force Model." *Mekatronika* 1 (1): 64–72.

Sharma, D., A. P. Bhondekar, A. Shukla, and C. Ghanshyam. 2018. "A Review on Technological Advancements in Crowd Management." *Journal of Ambient Intelligence and Humanized Computing* 9 (3): 485–495.

Shroff, P. H., and S. R. Chaudhary. 2017. "Critical Rendering Path Optimizations to Reduce the Web Page Loading Time." In *2017 2nd International Conference for Convergence in Technology (I2CT)*, 937–940. IEEE.

Singh, U., J.-F. Determe, F. Horlin, and P. De Doncker. 2020. "Crowd Forecasting Based on Wifi Sensors and LSTM Neural Networks." *IEEE Transactions on Instrumentation and Measurement* 69 (9): 6121–6131.

Specht, C., P. Dabrowski, J. Pawelski, M. Specht, and T. Szot. 2019. "Comparative Analysis of Positioning Accuracy of Gnss Receivers of Samsung Galaxy Smartphones in Marine Dynamic Measurements." *Advances in Space Research* 63 (9): 3018–3028.

Stieglitz, S., M. Mirbabaie, B. Ross, and C. Neuberger. 2018. "Social Media Analytics–challenges in Topic Discovery, Data Collection, and Data Preparation." *International Journal of Information Management* 39: 156–168.

Tricco, A. C., W. Zarin, E. Lillie, S. Jeblee, R. Warren, P. A. Khan, R. Robson, G. Hirst, and S. E. Straus. 2018. "Utility of Social Media and Crowd-intelligence Data for Pharmacovigilance: a Scoping Review." *BMC Medical Informatics and Decision Making* 18 (1): 38.

Waga, K., A. Tabarcea, R. Mariescu-Istodor, and P. Fränti. 2012. "System for Real Time Storage, Retrieval and Visualization of GPS Tracks." In *2012 16th International Conference on System Theory, Control and Computing (ICSTCC)*, 1–5. IEEE.

Waga, K., A. Tabarcea, R. Mariescu-Istodor, and P. Fränti. 2013. "Real Time Access to Multiple GPS Tracks." In *WEBIST*, 293–299.

Wirz, M., T. Franke, D. Roggen, E. Mitleton-Kelly, P. Lukowicz, and G. Tröster. 2012. "Inferring Crowd Conditions From Pedestrians' Location Traces for Real-Time Crowd Monitoring During City-Scale Mass Gatherings." In *2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 367–372. IEEE.

Wu, C., X. Ye, F. Ren, and Q. Du. 2018. "Check-in Behaviour and Spatio-temporal Vibrancy: An Exploratory Analysis in Shenzhen, China." *Cities (London, England)* 77: 104–116.

Xu, H., L. Li, and F. Fu. 2019. "Abnormal Behavior Detection Based on Spatio-Temporal Information Fusion for High Density Crowd." In *International Conference on Big Data Analytics for Cyber-Physical-Systems*, 1355–1363. Springer.

Yun, H. J., and M. H. Park. 2015. "Time–space Movement of Festival Visitors in Rural Areas Using a Smart Phone Application." *Asia Pacific Journal of Tourism Research* 20 (11): 1246–1265.

Zhao, X. 2015. "On Processing Gps Tracking Data of Spatio-temporal Car Movements: A Case Study." *Journal of Location Based Services* 9 (4): 235–253.

Zheng, Y., L. Capra, O. Wolfson, and H. Yang. 2014. "Urban Computing: Concepts, Methodologies, and Applications." *ACM Transactions on Intelligent Systems and Technology (TIST)* 5 (3): 1–55.