Journal of Ambient Intelligence and Smart Environments 13 (2021) 37–54
DOI 10.3233/AIS-200586
IOS Press

Forest path condition monitoring based on crowd-based trajectory data analysis

Francisco Arcas-Tunez and Fernando Terroso-Saenz *
Polytechnic School, Universidad Católica de Murcia (UCAM), Murcia, Spain
E-mails: farcas@ucam.edu, fterroso@ucam.edu

Abstract. The development of Road Information Acquisition Systems (RIASs) based on the Mobile Crowdsensing (MCS) paradigm has been widely studied for the last years. In that sense, most of the existing MCS-based RIASs focus on urban road networks and assume a car-based scenario. However, there exist a scarcity of approaches that pay attention to rural and country road networks. In that sense, forest paths are used for a wide range of recreational and sport activities by many different people and they can be also affected by different problems or obstacles blocking them. As a result, this work introduces SAMARITAN, a framework for rural-road network monitoring based on MCS. SAMARITAN analyzes the spatio-temporal trajectories from cyclists extracted from the fitness application Strava so as to uncover potential obstacles in a target road network. The framework has been evaluated in a real-world network of forest paths in the city of Cieza (Spain) showing quite promising results.

Keywords: Forest paths, crowdsensing, trajectory analysis

1. Introduction

For the last decade, smartphones have been the center of the digital life in modern societies due to their growing popularity. As a result, they are now equipped with several sensors like GPS, accelerometer, microphone, and so forth.

This palette of sensors allows to capture a large amount of contextual information related to the phone's holders and their surrounding environment [32]. In that sense, this contextual information has been used in several domains like tourism [2], assistance living [23] or personal training [14]. In addition to that, this has eased the development of the mobile crowdsensing (MCS) or human/people sensing paradigm. MCS allows to perceive large-scale phenomena that can not be detected at an individual level like the air pollution levels or the parking state of a city [5].

One of the most useful scenarios where MCS has been used is the deployment of innovative Road Information Acquisition Systems (RIAS). This type of systems report the condition of a road network like longitudinal and lateral roughness, friction, cracking or surface substance. This is instrumental information for road-maintenance operators. Traditionally, this information has been captured by means of infrastructure-based sensors like cameras or high-precision lasers [15]. In that sense, MCS allows to *enlarge* the coverage of this type of systems beyond the location of the infrastructure-based sensors by using the data captured by the position and motion sensors of the drivers' handheld devices [16,19,31]. The application domain of this MCS-based solutions restricts itself to urban road networks where different types of motor vehicles travels on [17].

However, existing literature has payed little attention to other types of road networks that are more common in a country environment like forest paths. These types of roads are widely used by people of all kinds for carrying out a large number of outdoor activities, such as cycling, running or going hiking. For that reason, they may suffer from condition problems that make them difficult to be properly used by visitors, like large obstacles (e.g. fallen trees) or landslides.

*Corresponding author. E-mail: fterroso@ucam.edu.

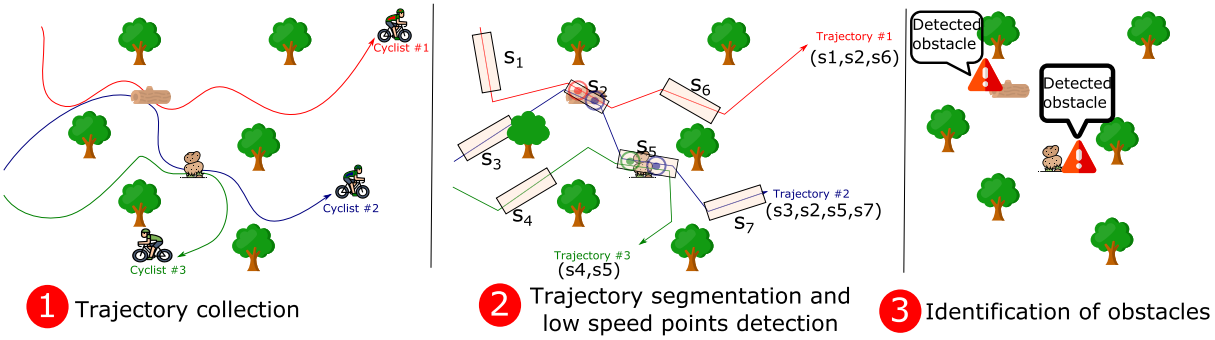


Fig. 1. Proposed methodology of SAMARITAN. The leftmost figure depicts the collection of the spatio-temporal trajectories of three cyclists moving around the region of interest. The central figure shows the mapping of the captured trajectories a to a set of seven community-based segments ($s_1 - s_7$), depicted as rectangles. The same figure shows a set of points where the system has detected an abnormal behaviour in terms of speed of trajectories #1 and #2 in segment s_2 and trajectories #2 and #3 in segment s_5. Finally, the rightmost figure shows the two alerts generated as system's outcome based on the aggregation of the previous abnormal points.

Due to their growing popularity, it becomes necessary the development of effective RIASs targeting rural or forest-related road networks. Nevertheless, existing solutions based on infrastructure sensors can not be applied in a cost-effective manner whereas MCS-based solutions rely on sensor-data analyses that focus on detecting fine-grained problems on asphalted roads. Nonetheless, this type of road is not the most common one in many rural environments.

Apart from that, we can also state the growing prominence of fitness apps, like Strava[1] or Endomondo[2] which operate on a large number of smartphones and wearable devices [25]. These applications allows to endlessly collect different features from exercise activities on large spatial areas. This makes these apps suitable enablers for crowd-based methods in a wide range of domains beyond the smart heath scope [12].

For that reason, the present work introduces SAMARITAN, a system for foreSt pAth Monitoring based on collAboRatIve Trajectory dAta aNalysis. The goal of SAMARITAN is to provide a RIAS for forest paths by following a MCS approach. In particular, it focuses on detecting obstacles that seriously affect the transit of people across such paths like landslides or fallen trunks. To do so, different well-known algorithms from the spatio-temporal trajectory-data mining field have been used.

As Fig. 1 depicts, SAMARITAN firstly collects the spatio-temporal trajectories of cyclists within a region of interest via the fitness-app Strava. Then, a two-level segmentation of the trajectories is applied. This step profits from the segments of interest defined by the own users in the Strava platform. This way, SAMARITAN leverages the knowledge shared by cyclists who move around the monitored spatial region. From the segments extracted of each individual trajectory, a set of candidates where a problem related to the paths state might occur is extracted based on the detection of abnormal *stop points*. Finally, the candidates extracted at individual level are aggregated to conform the final group of affected segments.

All in all, bearing in mind the open challenges of rural RIAS, the salient contributions of SAMARITAN are the following,

- First of all, it is the first crowd-based RIAS that fully focuses on the rural environment and its particularities.
- Secondly, it makes use of data extracted from a fitness application so as to uncover the state of a road network with great detail. In that sense, such extracted data does not limit to the raw spatio-temporal trajectories from the target users, but also the community-based segments that allow to guide the detection of incidents within the network.

The remainder of the paper is structured as follows. Next, an overview the relevant related work is put forward in Section 2. Section 3 is devoted to describing in detail the logic structure and the processing stages of SAMARITAN. Then, Section 4 discusses the main results of the performed experiments. Finally, the main conclusions and the future work are summed up in Section 5.

[1] https://www.strava.com/
[2] https://www.endomondo.com/

2. Related work

In this section we review the most recent advances in MCS-based RAISs along with the usage of data extracted from fitness applications as an enabler of innovative services in different environments.

2.1. MCS-based road information acquisition systems

During the last years, the ubiquity of smart devices carried by drivers has fostered the development of many different approaches for road state monitoring. In that sense, one the first proposals for MCS-based RAIS was put forward in [7]. In this work, authors designed a mechanism able to detect potholes and speedbumps via the analysis of accelerometer and GPS data. In [3] a privacy-preserving mechanism based on fog computing is defined so as to protect the road-condition reports sent by vehicles to the backend servers.

Nevertheless, a common feature of the aforementioned works is that they use onboard devices previously installed on vehicles instead of the drivers' handheld devices. This seriously limits their feasibility.

As a result, the work in [18] proposed a mechanism that actually profits from a smartphone equipment to detect road-surface problems. However, each device operates independently in the detection task so there is not a real cooperation among users. This makes rather difficult to achieve a complete coverage of a large road network.

This way, in [16] authors proposed an holistic architecture for MCS-based road status monitoring coined CRATER. This mechanism relies on different features extracted from the accelerometer of the users' smartphones so as to detect patholes and speedbumps via binary classifiers. Another interesting work is [17] where authors made use the crowdsensing platform *SmartRoadSense*, previously defined in [1], to perform a large-scale deployment of a RAIS as pilot study. Again, an accelerometer stream data is used to detect road-surface roughness. Besides, a time-series approach by means of the well-known algorithm Dynamic Time Warping for the detection of bumps or potholes is described in [28].

Another important research trend goes beyond the detection of single surface problems and intend to provide more general information about the road state. This is the case of [19] where authors follow a similar GPS and accelerometer fusion approach for road monitoring based on MCS. Unlike previous works they provide a more detailed solution to detect the general state of a road, not just surface problems like bumps or potholes. Similarly, a smartphone-based RAIS able to classify the quality of the road surface into five different levels ranging from *good* to *terrible* is put forward in [20].

Unlike these works, SAMARITAN focuses on rural roads that do not have the same problems as traditional urban roads. In our case, we detect obstacles that block certain rural paths. For that detection, we do not rely on measurements from the accelerometer sensor of smartphones like in the previous works. Instead of that, we analyze crowd-based GPS traces to detect abnormal *stay points* that might reflect some problems on certain path segments.

2.2. Analysis of fitness data

During the last years, many different applications have been developed by using the crowd-based data collected by sport and fitness applications.

One of the most interesting fields where this type of data has been applied is the analysis of human mobility flows in cities. For example, fitness-apps data is used in [27] to infer the usage patterns of certain recreational areas whereas in [22] a comprehensive study of mobility patterns based on Strava cycling data in Johannesburg (South Africa) is stated.

Moreover, an analysis based on Strava data of the correlation between certain characteristics of an urban road network and the volume of commute cycling is put forward in [12]. In [13], authors extend this type of analysis by including meteorological data in order to study how weather factors actually affect the behaviour of cyclists. Moreover, the work in [29] correlates the air-pollution level of a city with the mobility patterns of commuting and non-commuting cycling activities by using data from Strava. Similarly, authors in [9] make use of this type of data to detect cycling frequency behaviours in a city and in [10], a data-fusion approach, including data from Strava, is proposed to detect the points of interest where cyclists move within a spatial zone.

In our work, we use cycling data from Strava in a completely different scope. We make use of the spatio-temporal GPS trajectories extracted from that platform like the aforementioned works. These trajectories represent the routes taken by Strava users. However, we analyze the collected raw trajectories to detect points in a rural road network with potential obstacles. This

constitutes a novel usage of this type of fitness data beyond the mobility-pattern extraction mentioned above.

3. The SAMARITAN framework

In this section, we describe in detail the SAMARITAN framework. In that sense, Fig. 2 shows its key steps. As we can see, SAMARITAN follows a crowd-sensing approach where steps 2, 3 and 4 are executed in the contributors' devices whereas step 5 is executed in a backend server. The following subsections describe in detail each of these steps.

3.1. Collection of the trajectories

The first stage of the framework pipeline focuses on collecting the individual spatio-temporal trajectories of people moving around the target area of interest as Fig. 2 shows. For that goal, we make use of the Application Programming Interface (API) of the Strava platform.

We should remark that SAMARITAN focuses on the trajectories generated by cyclists instead of other sports. The rationale of this filtering is that the detection of problems in the road network is done by the analysis of abnormal speed fluctuations of the incoming trajectories. The cycling trajectories usually have a speed range large enough to perform such an analysis with an acceptable confidence level.

In our scope, a trajectory from a cyclist $c \in \mathcal{C}$ (where $\mathcal{C}$ is the set of target cyclists) represents the movement made by c during one route or trip with his bike. Therefore, a trajectory can be defined as follows,

Definition 1. A cyclist trajectory $tr^c \in \mathcal{TR}^c$ is a sequence of consecutive timestamped points of a cyclist c, $tr^c = \{p^c_{l_0,t_0} \rightarrow p^c_{l_1,t_1} \rightarrow \cdots \rightarrow p^c_{l_n,t_n}\}$, $n \geqslant 2$, where $p^c_{l_i,t_i} \in \mathcal{L} \times \mathcal{T}$ is the i-th location and timestamp of the trajectory so that $t_i < t_{i+1}\ \forall i \in [0, n]$.

All the trajectories collected from a particular cyclist c conform the set $\mathcal{TR}^c$. In that sense, as Fig. 2 shows, the SAMARITAN framework performs a set of analytical steps over each set of trajectories $\mathcal{TR}^c$ for all the target cyclists $\mathcal{C}$. These steps are labelled as 2a, 2b, 3 and 4 in the figure.

3.2. Segmentation of the trajectories

When it comes to analyze spatiotemporal trajectories, one common pre-procesing step is the *trajectory segmentation* [34]. This consists of dividing a trajectory into fragments by several criteria like time interval, spatial shape, semantic meaning. This allows to compress the incoming trajectory in a more simple format.

For this step, SAMARITAN makes use of the segments $\mathcal{S}$ defined by the own Strava community. In that sense, a Strava segment $s \in \mathcal{S}$ is just a portion of a

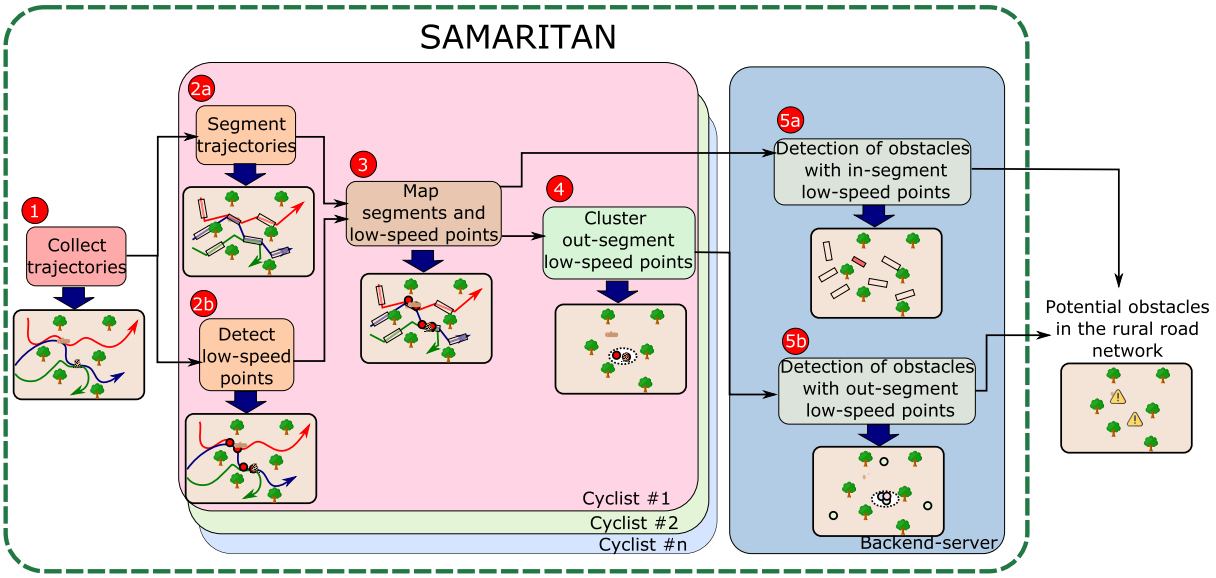


Fig. 2. Key steps of the SAMARITAN framework. The numbers in red reflect the execution order of each step.

road.[3] They are created by the own Strava users so that they can compare times marks at that particular road portion during their training.

Since each segment $s \in \mathcal{S}$ is defined as a line with a starting and ending point defined by coordinates that can be crawled via Strava API, it is possible to spatially join each trajectory $tr^c \in \mathcal{TR}^c$ with the set of community-based segments $\mathcal{S}$. Basically, this join takes each point $p^c_{l_i,t_i} \in tr_c$ and checks if it spatially fits into any of the line segments $s \in \mathcal{S}$.

As a result of this join, a new segment-based trajectory $tr^c_{\mathcal{S}}$ is generated from each point-based trajectory tr^c. It just comprises the Strava segments that contain any point $p^c_{l_i,t_i} \in tr_c$. This can be defined as follows,

Definition 2. A cyclist segment-trajectory $tr^c_{\mathcal{S}} \in \mathcal{TR}^c_s$ is a sequence of segments, $tr^c_{\mathcal{S}} = \{s^c_0 \rightarrow \cdots \rightarrow s^c_m\}$, $m \geqslant 1$, where $s^c_i \in \mathcal{S}$ is the i-th segment of the trajectory.

For example, in Fig. 1, the trajectory of cyclist 1, tr^1, gives rise to the trajectory $tr^1_s = \langle s_1 \rightarrow s_2 \rightarrow s_6 \rangle$. In case of the trajectory of cyclist 2, tr^2, it generates the trajectory $tr^2_s = \langle s_3 \rightarrow s_2 \rightarrow s_5 \rightarrow s_7 \rangle$. Therefore, we can see that the number of segments of each trajectory might be different in each case.

3.3. Detection of low-speed points

Apart from the segmentation described above, we also analyze each incoming raw trajectory tr^c so as to detect the parts where the cyclist c moved quite slowly (step 2b in Fig. 2).

The rationale of detecting these low-speed parts of a trajectory is that they might indicate the presence of certain obstacles during a cyclist's trip. In these points, a cyclist usually needs to slow down and, in some cases, even get off his bike. This is reflected as a sudden drop in the speed profile of the trajectory. These low-speed points can be regarded as outliers in the speed time-series of a trajectory.

This speed time-series can be calculated from the sequence of timestamped points of a raw trajectory stated in Def. 2. In particular, for each pair of consecutive points $\langle p^c_{l_{i-1},t_{i-1}}, p^c_{l_i,t_i} \rangle \in tr^c$, we compute its instantaneous speed as $sp^c_i = \frac{\text{dist}(l_{i-1},l_i)}{t_i - t_{i-1}}$ where $\text{dist}(l_{i-1}, l_i)$ refers to the Haversine distance between the two points. This gives raise to the associate speed time-series of the trajectory that is defined as follows,

[3] https://support.strava.com/hc/en-us/articles/216918167-Strava-Segments

Definition 3. A cyclist trajectory speed profile tr^c_{sp} is a sequence of speed measurements from a trajectory, $tr^c \in \mathcal{TR}^C$, $tr^c_{sp} = \langle sp^c_1, \ldots, sp^c_o \rangle$, $o \geqslant 1$, where sp_i is the instantaneous speed at the i-th location of tr^c.

In order to detect the outliers from this profile, SAMARITAN makes applies the Generalized Extreme Student Deviation (GESD) test over the time-series tr^c_{sp} [26]. GESD is a simple but robust procedure to detect outliers in time series in many different situations based on the GESD test [11]. In brief, the GESD infers whether a set of extreme observations of a time series are actually outliers based on their associated t-distribution.

In order to only retain the low-speed outliers, we discard those abnormal values that are above 0.5 m/s. As a result of this process, a set of abnormally low speed values $tr^c_{sp,\text{out}} \subset tr^c_{sp}$ are extracted.

Finally, we map each abnormal speed $sp^c_{i,\text{out}} \in tr^c_{sp,\text{out}}$ to its corresponding point $p_{l_i,t_i} \in tr^c$. This gives raise to the set of points $tr^c_{ls} \subset tr^c$ where such abnormal low speeds occurred.

It is worth-mentioning that the aforementioned procedure relies on GPS trajectories that are defined at a very fine granularity where the current location of the cyclist is captured every few seconds. The functionality of SAMARITAN would be rather limited if the GPS feed (e.g. the cyclist's smartphone or smartwatch) was configured with a large sampling rate because, in that case, some low-speed points might not be detected.

However, high-intensity sports are better monitored when high sampling rates above 10 Hz are used [24]. Furthermore, some manufacturers of GPS trackers in the sport field actually recommend a similar configuration for their devices [30]. Consequently, it is sensible to expect that the potential contributors of SAMARITAN would generate fine-grained trajectories during its cycling activities in most of the cases.

3.4. Mapping of low-speed points and segments

Once we have uncovered the abnormal low-speed points of a trajectory, we need to detect whether these points occurred or not in any of the community-based segments of the trajectory. This is because SAMARITAN handles the points in a different manner depending on they fit or not into a Strava segment as we will see later.

Consequently, this step of SAMARITAN takes as input the Strava segments $tr^c_{\mathcal{S}}$ and the set of low-speed

Algorithm 1: Pseudo-code of the low-speed points mapping

Input: low-speed points of the trajectory tr^c_{ls}, community-based segments of the trajectory $tr^c_{\mathcal{S}}$.
Output: low-speed points in community-based segments tr^c_{ISLSP}, low-speed points outside community-based segments tr^c_{OSLSP}

1 $tr^c_{\text{ISLSP}} \leftarrow tr^c_{\text{OSLSP}} \leftarrow \emptyset$
2 **for *each*** $p_{l,t} \in tr^c_{ls}$ **do**
3 $s_p \leftarrow \emptyset$
4 **for *each*** $s \in tr^c_{\mathcal{S}}$ **do**
5 **if** include$(p_{l,t}, s)$ **then**
6 $s_p \leftarrow s$
7 **if** $s_p \neq \emptyset$ **then**
8 $tr^c_{s,t} \leftarrow \text{time}(tr^c, s_p)$
9 $z_t \leftarrow \frac{tr^c_{s,t} - \mu_t^{s_p}}{\sigma_t^{s_p}}$
10 **if** $z_t \geqslant z_{\max}$ **then**
11 $tr^c_{\text{ISLSP}} \leftarrow tr^c_{\text{ISLSP}} \cup p_{l,t}$
12 **else**
13 $tr^c_{\text{OSLSP}} \leftarrow tr^c_{\text{OSLSP}} \cup p_{l,t}$
14 **return** $tr^c_{\text{ISLSP}}, tr^c_{\text{OSLSP}}$

points tr^c_{ls} of each incoming trajectory tr^c (see Fig. 2). Algorithm 1 summarizes the mapping processing perform at this stage of the framework.

As we can see from the pseudo-code, we just take each low-speed point in tr^c_{ls} (line 2) and check whether it spatially fits into in any of the segments of $tr^c_{\mathcal{S}}$ (lines 4–6).

If the point fits into a segment then we perform a time-based analysis (lines 7–11 of Alg. 1). The idea of this analysis is that if a cyclist had to abruptly slow down one or more times within a segment then the time required to cover that segment would be meaningfully larger than the cyclist's average for that particular segment. Otherwise, the detected low-speed points may be just noisy measurements.

To do so, we profit from a feature of the Strava API[4] that allows to extract the historic time records of a cyclist for a given segment. Hence, we can compute the mean and the standard deviation of these time records for a cyclist c in any segment $s \in \mathcal{S}$, $\langle \mu_t^{s,c}, \sigma_t^{s,c} \rangle$.

Apart from that, we can calculate the actual time required by a cyclist to cover a segment s during a trajectory tr^c ($tr^c_{s,t}$). Given that value, we can estimate its z-score in the Gaussian Distribution $\mathcal{N}(\mu_t^{s,c}, \sigma_t^{s,c})$ (line 9). This score indicates how many standard deviations $\sigma_t^{s,c}$ the mark $tr^c_{s,t}$ is far away from the mean $\mu_t^{s,c}$.

If this z-score is above a certain threshold ($z_{\max}$) then we can conclude that the target cyclist has not only abruptly decelerated within the segment (because of the existence of low-speed points), but has also moved much lower than usual. As this can be regarded as a truly abnormal behaviour, the target low-speed point is included as part of the set of in-segment low-speed points (ISLSPs) tr^c_{ISLSP} for further analysis (line 11).

Otherwise, if the low-speed point does not fit into any segment we can not perform the aforementioned time analysis. This type of out-segment points are handled in a different manner. Thus, they are included in the complementary set of out-segment low-speed points (OSLSPs) tr^c_{OSLSP}. Consequently, we can see that the following two conditions hold,

- $\langle tr^c_{\text{ISLSP}} \cap tr^c_{\text{OSLSP}} \rangle = \emptyset$
- $\langle tr^c_{\text{ISLSP}} \cup tr^c_{\text{OSLSP}} \rangle = tr^c_{ls}$

Finally, these two sets are processed in a different way by SAMARITAN. This is because both sets represent completely different situations. Whilst the low-speed points in tr^c_{ISLSP} occurred in regions (segments) with a high volume of cyclists, tr^c_{OSLSP} include the low-speed points that occurred in spatial regions that *might* not be heavily traveled by cyclists.

3.5. *Clustering of the out-segment low-speed points (OSLSPs)*

As we have mentioned before, tr^c_{OSLSP} contains the OSLSPs of a trajectory tr^c, that is, the locations where the cyclist moved abnormally and suddenly slow in spatial areas not covered by any community-based segment $\mathcal{S}$.

In this case, we need to determine if these abrupt decelerations correspond to recent changes in the mobility behaviour of cyclist c or they are just part of his usual behaviour. This because a sudden change in the mobility profile of a cyclist in terms of new *stop points* might indicate the presence of recent obstacles in these points.

[4] http://developers.strava.com/docs/reference/#api-SegmentEfforts-getEffortsBySegmentId

Algorithm 2: Pseudo-code of OSLSPs clustering

Input: set of OSLSPs $\mathcal{P}^c_{\text{OSLSP}}$, distance threshold ϵ, minimum number of points threshold *minPoints*, maximum time interval Δ_t

Output: recent low-speed centroids ls^c

1 $\mathcal{LS}^c \leftarrow \emptyset$
2 $\mathcal{CP}^c \leftarrow$ DBSCAN($\mathcal{P}^c_{\text{OSLSP}}$, ϵ, *minPoints*)
3 **for** ***each*** $cp \in \mathcal{CP}^c$ **do**
4 $\quad t_{\text{min}} \leftarrow \min_t(\mathcal{N}(cp)_\epsilon)$
5 $\quad$ **if** $|t_{\text{now}} - t_{\text{min}}| \leqslant \Delta_t$ **then**
6 $\quad\quad \mathcal{LS}^c \leftarrow \mathcal{LS}^c \cup cp$
7 **return** $\mathcal{LS}^c$

In order to determine if the OSLSPs are part of the usual mobility behavior of a cyclist or not, we follow a density-based clustering approach based on the well-known DBSCAN algorithm [8]. This way, we collect all the OSLSPs from each cyclist c during the last $\Delta^{\mathcal{C}}_{\text{OSLSP}}$ hours. This gives raise to the set $\mathcal{P}^c_{\text{OSLSP}}$.

Based on the OSLSPs included in that dataset, we applied the procedure described in Algorithm 2. First of all, we apply the density-based clustering algorithm DBSCAN [8] to $\mathcal{P}^c_{\text{OSLSP}}$ (line 2 of Alg. 2). The key steps of DBSCAN can be summarized as follows,

- For each OSLSP $p_{l,t} \in \mathcal{P}^c_{\text{OSLSP}}$, the algorithm detects the other low-speed points located within the distance ϵ from it. This is its ϵ-neighborhood $\mathcal{N}(p_{l,t})_\epsilon$.
- If the number of points within that neighborhood is above the *minPoints* then the $p_{l,t}$ is considered a centroid and all the points in its ϵ-neighborhood belong to the same cluster.

The set of centroids obtained from this algorithm ($\mathcal{CP}^c$) indicates the locations where a cyclist c has abruptly slow down several times during their routes for the last $\Delta^{\mathcal{C}}_{\text{OSLSP}}$ hours. However, we need to check, for each location, if this is a usual behavior or a recent one.

To do so, we extract the minimum timestamp t_{min} of all the points included in the ϵ-neighborhood ($\mathcal{N}(cp)_\epsilon$) of a centroid cp (lines 3–4). If the difference between that timestamp and the current one (t_{now}) is below a certain threshold Δ_t ($< \Delta^{\mathcal{C}}_{\text{OSLSP}}$) (line 5), this indicates that all the points in the ϵ-neighborhood of target centroid cp have been generated quite recently. Therefore, this point is included as a representative out-segment low-speed point (r-OSLSP) in the set $\mathcal{LS}^c$ (line 6).

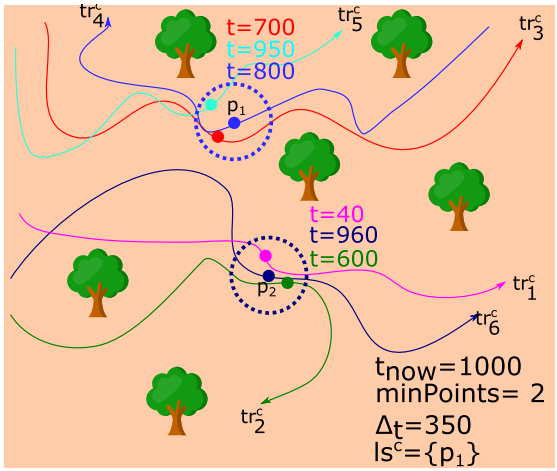


Fig. 3. Example of detection of a r-OSLSP p_1 based on six different trajectories from cyclist c. Each trajectory comprises a single OSLSP depicted as a colored dot.

An illustrative example of this process is shown in Fig. 3. In this scenario, six different trajectories from the same cyclist c are collected, each one comprising a single OSLSP. Next, SAMARITAN executes Algorithm 2. In this case, $\mathcal{N}(p_1)_\epsilon$ comprises 2 different OSLSPs (from trajectories tr^c_3 and tr^c_4). As this number is equal to *minPoints*, the time interval calculation of the ϵ-neighborhood is performed. In this case, its t_{min} is 700 time units. Since $t_{\text{now}} - t_{\text{min}}$ $(1000 - 700 = 300)$ is below Δ_t then p_1 is included in the ls^c set.

A similar situation arises with trajectory tr^c_6 and its low-speed point p_2. However, in this case the t_{min} of its ϵ-neighborhood takes 40 time units as value. This makes that the covered time interval of the neighborhood $(1000 - 40 = 960)$ is much longer than the Δ_t. This excludes p_2 from ls^c as its coverage area includes stop points that occurred a long time ago.

As Fig. 2 depicts, we should mention that the four procedures described in Sections 3.2 to 3.5 are executed independently for each cyclist in $\mathcal{C}$.

At this point, we should indicate that it is true that GPS trajectories usually suffer from some inaccuracies due to signal-reception problems. They might cause that the path represented by a trajectory does not completely fit the actual path followed by the moving object. However, this type of error usually arises in scenarios where signal occlusion occurs like indoor environments or certain urban regions [33].

In that sense, the present framework focuses on an outdoor scenario that may reduce the incidence of this

type of error. Moreover, the low-speed point extraction does not rely on the actual *trace* followed by a cyclist but on his speed evolution. It is true that inaccurate GPS trajectories might cause the system to wrongly infer false low-speed points or not to detect certain true ones. However, the inference of ISLSPs and OSLSPs depends on the latent speed of a trajectory at Strava segment level (for ISLSPs) or considering a ϵ-neighborhood (for OSLSPs). This segment or cluster matching procedure is similar to the map-matching step performed by many solutions for GPS trajectory processing in the urban and vehicular environment [4,6,21]. This allows to reduce the impact of these noisy points in the speed pattern of a trajectory. Hence, this makes it possible to detect that a cyclist has moved abnormally slow or not.

Finally, the tr^c_{ISLSP} and tr^c_{OSLSP} points individually generated by each cyclist are send to a back-end server to further processing them in an aggregated manner.

3.6. Detection of obstacles with in-segment low-speed points (ISLSPs)

Based on the procedure described in Section 3.4, the client-side of SAMARITAN was able to extract the set tr^c_{ISLSP} with the ISLSPs of each trajectory $tr^c \in \mathcal{TR}^c$. Given this set, the back-end server of the framework can now detect if there is any anomaly in the cyclists' mobility behavior in any of the Strava segments $\mathcal{S}$.

To do so, SAMARITAN follows a batch-based analysis (depicted as step 5a in Fig. 2). To begin with, it defines a time-based sliding window that collects all the sets tr^c_{ISLSP} generated by all cyclists $\mathcal{C}$ during the last Δ_{ISLSP} hours. The content of this sliding-windows is defined as the set $\mathcal{W}_{\text{ISLSP}}$. This set contains all the ISLSPs occurred in the most recent trajectories received by SAMARITAN.

Given the aforementioned dataset, SAMARITAN detects changes in the mobility behavior within each Strava segment by following the procedure described in Algorithm 3.

Basically, we select for each segment $s \in \mathcal{S}$ the ISLSPs that fit in it (lines 2–3 of the algorithm). Next, we launch an instance of the DBSCAN algorithm with the selected subset (line 4). The generated centroid are considered potential obstacles and aggregated to the $\mathcal{O}^{\mathcal{S}}$ set (line 5).Finally, Algorithm 3 is launched each time a new tr^c_{ISLSP} is generated by any cyclist $c \in \mathcal{C}$.

An alternative approach for this step would have been the execution of a global instance of DBSCAN independent of any segment. Nonetheless, this had merged together ISLSPs of different segments. Since each Strava segment conceptually represents a particular road slice with different features it is necessary to analyze each segment independently.

All in all, we can see that the Strava segments provides a sparse spatial tessellation of the area under study. They allow to group together stop-points in spatial areas frequently crossed by cyclists.

Algorithm 3: Pseudo-code of the detection of potential obstacles based on ISLSPs

Input: set of ISLSPS W_{ISLSP}, set of segments $\mathcal{S}$, distance threshold ϵ, minimum number of points threshold *minPoints*
Output: set of potential obstacles' location $\mathcal{O}^{\mathcal{S}}$

1 $\mathcal{O}^{\mathcal{S}} \leftarrow \emptyset$
2 **for** ***each*** $s \in \mathcal{S}$ **do**
3 $\quad \mathcal{W}^s_{\text{ISLSP}} \leftarrow \text{select}(\mathcal{W}_{\text{ISLSP}}, s)$
4 $\quad \mathcal{CP}_{\text{ISLSP}} \leftarrow \text{DBSCAN}(\mathcal{W}_{\text{ISLSP}}, \epsilon, \textit{minPoints})$
5 $\quad \mathcal{O}^{\mathcal{S}} \leftarrow \mathcal{O}^{\mathcal{S}} \cup \mathcal{CP}_{\text{ISLSP}}$
6 **return** $\mathcal{O}^{\mathcal{S}}$

3.7. Detection of obstacles with out-segment low-speed points (OSLSPs)

In Section 3.5 we put forward how uncover the r-OSLSPs from each cyclist under control. Therefore, we can use that information so as to detect a new set of obstacles apart from the ones uncovered by the procedure described in the previous section based on ISLSPs.

As in Section 3.6, we analyze the collected r-OSLSPs by making use of a batch-based approach. In particular, we gather the r-OSLSPs from all the cyclist in $\mathcal{C}$ generated during the last $\Delta_{\text{r-OSLSP}}$ hours by means of a time-based sliding window $\mathcal{W}_{\text{r-OSLSP}}$.

Next, each time $\mathcal{W}_{\text{r-OSLSP}}$ is updated with a new r-OSLSP set, $\mathcal{LS}^c$, DBSCAN is applied on the new sliding-window content. The two parameters of the algorithm (*minPoints* and ϵ) are set with the same values that the instance of DBSCAN used to uncover the personal r-OSLSPs in Section 3.5.

The resulting set of centroids indicate spatial regions (not covered by a Strava segment) where several cyclist had to abruptly decelerate during, at least, the last $\Delta_{\text{r-OSLSP}}$ hours. Therefore, this set of centroids is defined as the second type of potential obstacles $\mathcal{O}^{\text{in}\,\mathcal{S}}$ detected by SAMARITAN.

We can see that, in order to process, the OSLSPs SAMARITAN follows a two-level DBSCAN clustering.

- In the first level, OSLSPs are clustered together for each individual cyclist as put forward in Section 3.5.
- In the second level, all the r-OSLSPs extracted in the first clustering level are aggregated and clustered again so as to come up with the final set of candidate locations of obstacles ($\mathcal{O}^{\text{in }\mathcal{S}}$).

This approach makes SAMARITAN able to be adapted to a client-server infrastructure where the first clustering level is executed in the cyclists's personal devices whereas the second one is executed in a back-end server with the meaningful stops from the contributors.

Finally, the set of obstacles $\mathcal{O}^{\text{in }\mathcal{S}}$ and $\mathcal{O}^{\mathcal{S}}$ are merged together as the final set of obstacles identified by SAMARITAN. In that sense, we should remark that the two sets are disjoined because $\mathcal{O}^{\mathcal{S}}$ reports obstacles within Strava segments whereas $\mathcal{O}^{\text{in }\mathcal{S}}$ are obstacles in parts of the rural-road network not identified by any community-based platform.

3.8. Summary of the SAMARITAN pipeline

For the sake of clarity, here we sum up the key steps that compose the processing pipeline of the SAMARITAN framework as shown in Fig. 2.

First of all, the client-side SAMARITAN collects the spatio-temporal trajectories of its target cyclist moving around the spatial area under monitoring (step 1 in Fig. 2). Then, these trajectories are split based on the community-defined Strava segments and their low-speed points are uncovered (steps 2a and 2b). Next, these low-speed points are mapped based on the Strava segments (step 3). This gives raise to OSLSPs and ISLSPs. After that, the OSLSP are clustered so as to uncover the r-OSLSPs (step 4).

Finally, the central server of the framework merges together the ISLSP in each Strava segment to detect the potential obstacles within these segments (step 5a). At the same time, the r-OSLSPs are clustered with DBSCAN to identify regions with potential obstacles located outside any Strava segment (step 5b).

3.9. Data privacy aspects of SAMARITAN

Regarding the issues about the cyclists' privacy when using SAMARITAN, it is true that the proposed solution relies on the analysis of the spatio-temporal trajectories from cyclists so as to detect forest-path incidents. However, the initial processing of the raw GPS traces is performed locally in the client side of the framework running in each cyclist's mobile device. This client side only sends to the central server the low-speed points tr^{c}_{ISLSP} and tr^{c}_{OSLSP} that have been uncovered, not the whole spatio-temporal trace of the trajectory (see Sections 3.4 and 3.5).

Consequently, the location data sent to the central server is actually quite limited. In addition to that, the server only needs to store the low-speed points sent by the cyclist during a certain amount of time due to its batch-based computation of the aggregated points (see Sections 3.6 and 3.7). After Δ_{ISLSP} and $\Delta_{\text{r-OSLSP}}$ hours, which are the length of the time-based sliding windows used for such an aggregation, the cyclists' reports can be removed from the server. Since users only report sparse data and it is not stored in the server for a long period of time, this would make it rather difficult to uncover private or personal information from the contributors, like their home location, by using solely the information contained in the SAMARITAN central server.

For the sake of completeness, the privacy-preserving solution stated in [3] focuses on a scenario where vehicles are endlessly reporting real-time data to certain intermediate entities called Roadside Units (RSUs). These units analyze the fine-grained data from vehicles and report possible alerts about road conditions to upper cloud servers. Authors focus on developing a mechanism to secure the communication channel between vehicles and RSUs as a large amount of private data flows through it. On the contrary, SAMARITAN does not require contributors to send real-time data to a central or intermediary server but certain candidate points where a road problem might occur. Therefore, the direct integration of the aforementioned privacy solution in SAMARITAN would not be possible.

3.10. Configuration of SAMARITAN

The proposed framework is able to control its *sensibility* to detect forest-path problems. This is because this detection mainly depends on two different sets of parameters, the lengths of the time-windows (Δ_{ISLSP}, $\Delta_{\text{r-OSLSP}}$) used to store the most recent low-speed points from all the cyclists and the DBSCAN parameters (*minPoints*, ϵ), which define the density of these points required to infer that there is a particular problem in a forest path.

Fig. 4. Spatial location of the Strava segments used in the evaluation. Each segment is depicted as a coloured line.

These sets of parameters allow to adjust the behaviour of SAMARITAN with respect to the number of available cyclists using the service. Depending on the number of contributors, the SAMARITAN administrator can reduce or increase the length of the time windows by considering the storage capability of the central server.

In case of the number of contributors is high, the length of the time-windows can be reduced. This is because provided that there is a problem in a forest-path many cyclists would report abnormal low-speed points in a short period of time.

On the contrary, if the number of contributors is low, the administrator can reduce the density-parameters of the DBSCAN algorithm so as to increase the sensibility of the solution. It is true that this might lead to reporting some false positives but, in this case, the important goal is to detect as many obstacles as possible.

4. Evaluation of the framework

We have evaluated SAMARITAN in the city of Cieza at the southwest of Spain. This small city is surrounded by many forest paths that are very popular among local cyclists. Furthermore, the region was heavily affected by a cold front in September of 2019.[5] This caused a lot of damage to the roads in the area.

[5] https://english.elpais.com/elpais/2019/09/16/inenglish/1568618372_639259.html

Table 1
Length parameters of the Strava segments

Segment feature	Value
Min. segment length	92 m
Max. segment length	1922 m
Avg. segment length	673 m
Total segment length	24228 m

In order to apply SAMARITAN in that region we focused on the city outskirts included the spatial bounding box defined by the latitude-longitude coordinates (38.203332, −1.468835) and (38.234046, −1.397972).

4.1. Data description

The details of three datasets used for the present evalution are described next.

4.1.1. Strava segments under consideration

The target geographical area includes 36 Strava segments whose spatial distribution within the target bounding box is shown in Fig. 4. As we can see, these segments are not homogeneously distributed in the target area. This is because they are defined by the own Strava users. Therefore, they usually represent parts of the road that are interesting or event challenging for the cyclist perspective.

Besides, Table 1 indicates the minimum, average and maximum length of these segments. We can see that these segments cover a wide range of lengths from a few meters to almost 2000 meters.

Furthermore, Fig. 5 shows the length distribution among the collected Strava segments. From this distribution, we can see that most of the segments cover road parts of roughtly 317 meters.

This variety of lengths justifies the step take by SAMARITAN to identify obstacles within Strava segments described in Section 3.6. As we saw in that section, the ISLSPs of a particular segment are clustered by means of DBSCAN. This allows to cope with quite large segments as the framework is able to provide different tentative locations within a segment.

4.1.2. Spatio-temporal trajectories under study

Regarding the trajectories used as input by the framework, we have collected the data from 5 different cyclists during a six-month period from 01/06/2019 to 31/12/2019. This has given raise to 98 different spatio-temporal trajectories. The sampling rate of these trajectories varied from 1 s to 3 s.

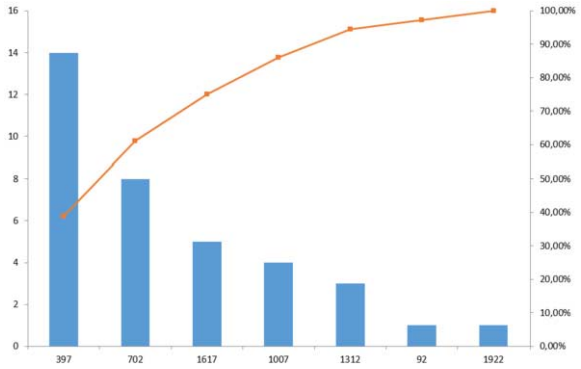


Fig. 5. Histogram with the length distribution of the Strava segments. The x-axis shows the segment length and the y-axis indicate the number of segments. The red line shows the cumulative percentage of segments.

4.1.3. Target obstacles

In order to test the accuracy of SAMARITAN, eight particular locations within the target geographical area where a clear alteration of a forest path occurred were used as ground truth. They were identified as $\mathcal{O}_{\text{true}} = \langle o_1, o_2, \ldots, o_8 \rangle$. Figure 6 shows the position of these obstacles. As we can see, seven of them fit into a Strava segment whereas one was outside any of them.

All these obstacles were caused by a three-day cold front that occurred during the 11st and 14th of September of 2019 in the southeast of Spain. This storm caused torrential downpours that seriously affected the spatial region considered by the use case. As a result, five of the target problems were caused by landslides, three by fallen trees and one by a broken pipe. All these obstacles were manually discovered by the authors during a three-day campaign from 27th to 29th of September. In that sense, this ground-truth information was not revealed to the 5 cyclists acting as contributors of SAMARITAN.

For the sake of clarity, Table 2 shows the relationship between the aforementioned obstacles and some detail of the segments comprising them. Furthermore, Fig. 7 shows some of these obstacles. As we can see, they were caused by landslides (obstacle o_1), fallen trees (o_2 and o_7) and broken irrigation channels (o_4).

Fig. 6. Location of the eight target obstacles. Each obstacle is depicted as a red point. Obstacles $o_1 - o_7$ fit into a Strava segment whereas o_8 was outside any segment.

Table 2

Distribution of the in-segment target obstacles. The *segment name* column indicates the name of the segment according to the Strava API

Segment name	Segment length	Obstacle id.
Bajada Senda del Moro (s_1)	1500 m	o_1
		o_2
Menu-Puente Abaram (s_2)	1900 m	o_3
		o_4
Subida Camino-Viejo (s_3)	1100 m	o_5
		o_6
		o_7

(a) Obstacle o_1: Landslide

(b) Obstacle o_2:Fallen tree I

(c) Obstacle o_4: Broken irrigation canal

(d) Obstacle o_7: Fallen tree II

Fig. 7. Visual inspection of the detected obstacles.

Table 3

SAMARITAN settings

Parameter	Step	Value
$z_{\max}$	Low-speed points mapping (Section 3.4)	1.5
ϵ	OSLSP clustering (Section 3.5)	250 m
minPoints		2
$\Delta^{\mathcal{C}}_{\text{OSLSP}}$		720 h
Δ_t		360 h
Δ_{ISLSP}	Obstacle detection with ISLSPs (Section 3.6)	360 h
$minPoints_{ls}$		4
$\Delta_{\text{r-OSLSP}}$	Obstacle detection with r-OSLSPs (Section 3.7)	360 h

4.2. Framework parameters

For the sake of clarity, Table 3 sums up the parameter settings in this use case.

As we can see, the parameters that define the time periods to analyze the low-speed points (Δ_t, Δ_{ISLSP}, $\Delta_{\text{r-OSLSP}}$) are set to 360 hours (2 weeks). This is because it allows to collect enough data so as to detect obstacles in a more reliable manner.

This is because SAMARITAN relies on routes made by cyclist during their free time, so they can not be regarded as daily trips. This makes the required time periods to process data quite large.

4.3. Analysis of the results

Given the trajectories and segments described above, SAMARITAN reported 10 different potential obstacles $\mathcal{O}^{\text{rep}} = \langle o_1^{\text{rep}}, o_2^{\text{rep}}, \ldots, o_{10}^{\text{rep}} \rangle$ as shown in Fig. 8.

From this set, seven of them (o_3^{rep}, o_5^{rep}, o_6^{rep}, o_7^{rep}, o_8^{rep}, o_9^{rep} and o_{10}^{rep}) were located less than 150 meters of a ground truth alarm. In that sense, Table 4 shows the closest true obstacle for each of the generated obstacles.

In order to properly analyze these results, we study the speed behaviour of the trajectories in the three segments comprising seven of the true obstacles (s_1, s_2 and s_3).

Furthermore, some of these obstacles were repaired by authorities during the time period of the present use case. This allowed us to split such speed profile in three different time intervals, 1) one including the trajectories before the occurrence of the obstacle event, 2) another including the trajectories in the time interval during which the obstacle was present and 3) a final time interval covering the trajectories after the forest-path problem was solved.

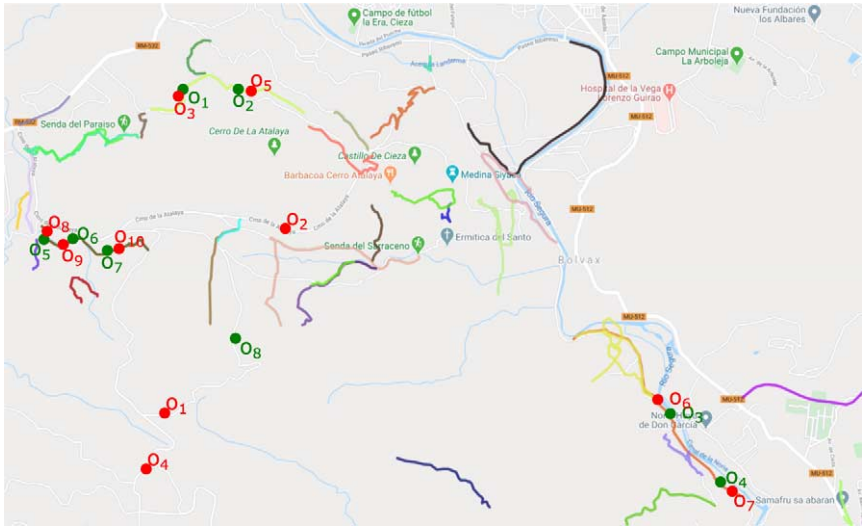


Fig. 8. Location of the ten obstacles detected by SAMARITAN. Each obstacle is depicted as a red point whereas the true obstacles are shown in green.

Table 4
Relation between the inferred obstacles by SAMARITAN and the true ones along with the distance between each pair of inferred and true points. The rows in grey indicate distances below 150 m

Inferred obstacle	Closest true obstacle	Distance
o_1^{rep}	o_8	1133 m
o_2^{rep}	o_8	1344 m
o_3^{rep}	o_1	65 m
o_4^{rep}	o_8	1500 m
o_5^{rep}	o_2	87 m
o_6^{rep}	o_3	81 m
o_7^{rep}	o_4	130 m
o_8^{rep}	o_5	37 m
o_9^{rep}	o_6	26 m
o_{10}^{rep}	o_7	91 m

To set the date thresholds defining each interval, we used the average time marks of the five target cyclist in each segment depicted in Fig. 9. As we can see, there is a clear increment of the time marks in the three segments after the 8th of September of 2019. This consistent with the dates of the cold front affecting the region in that month.

After that, the marks behavior varies depending on the segment. In segment s_1 (Fig. 9a) the marks go back to the values before the cold front after roughly two months. In segment s_2 (Fig. 9c) we can see slight decrease of the marks around the 20th of October. Finally, segment s_3 does not exhibit any decrements after the cold front (Fig. 9e).

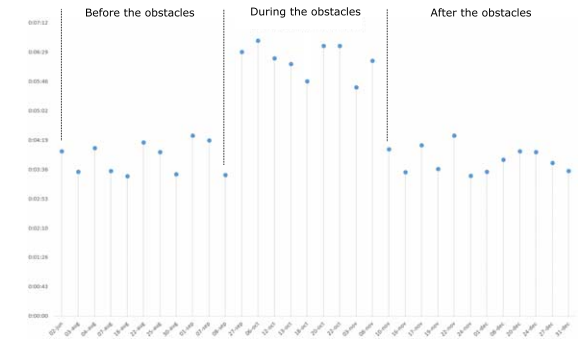
Besides, the speed distributions shown in Figs 9b,d&f confirm that the presence of obstacles clearly affects the speed behaviour of cyclists around the affected region.

Figure 10 shows the average speed evolution of the target trajectories in segments s_1, s_2 and s_3 during these three time intervals. The figure also includes the location of the inferred and true obstacles with respect the length of the segment. For example, obstacles o_1 is at 630 m from the beginning of segment s_1 (see Fig. 10a).

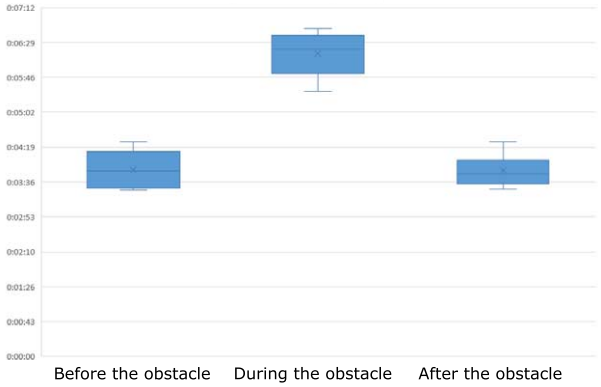
The first thing to note is that there is a clear difference between the trajectories' speed profile depending on the time period. This shows that the presence of obstacles meaningfully affects the mobility flows of cyclist moving around the area under control.

More in detail, we can see that the speed-profile after the obstacle events and before their repair are quite similar in segment s_1 (Fig. 10a), indicating that the forest path in that segment was fully recovered. However, the speed profiles before and after the obstacle event in segment s_2 (Fig. 10b) are quite different. In particular, cyclist have now to move much more slowly than before the obstacle event. Finally, in segment s_3 there was not any repair intervention. Hence, the speed profiles of the trajectories during and after the obstacle event are quite similar.

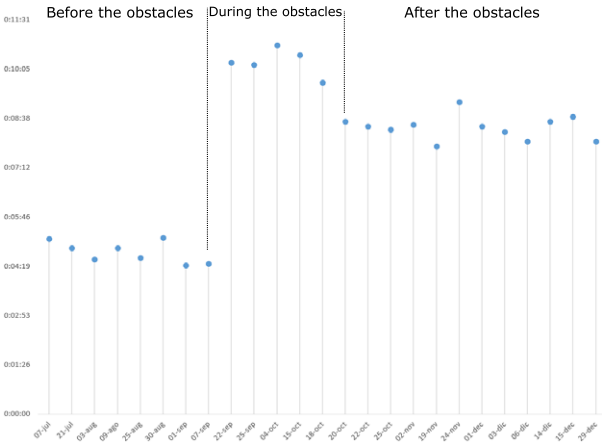
If we focus on the speed behavior of the trajectories during the obstacle presence (yellow lines in Fig. 10), we can clearly see a set of sudden speed drops around the obstacles location. This generates different ISLSPs



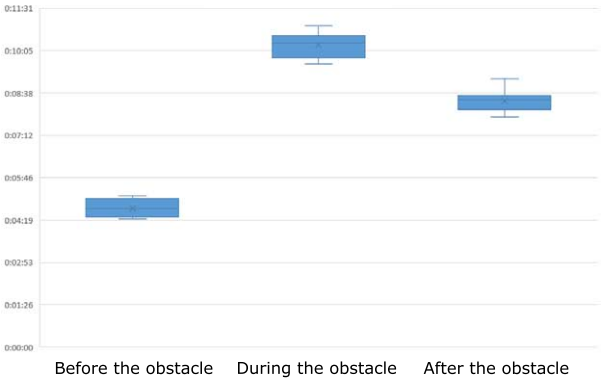
(a) Segment s_1: Evolution of the average time marks



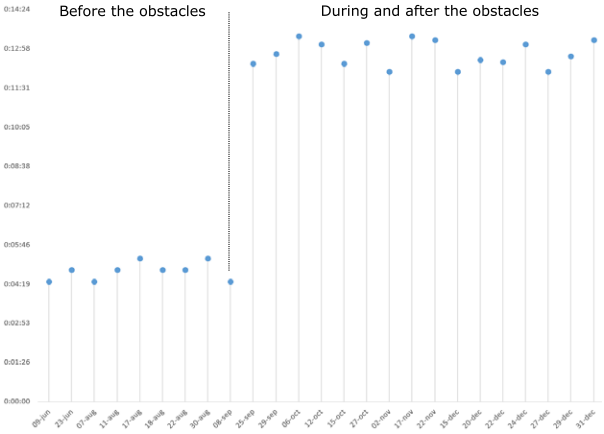
(b) Segment s_1: Distribution of the time marks per time period.



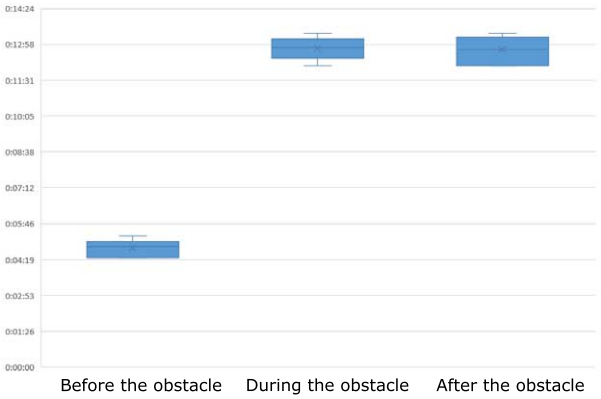
(c) Segment s_2: Evolution of the time marks



(d) Segment s_2: Distribution of the time marks per time period.

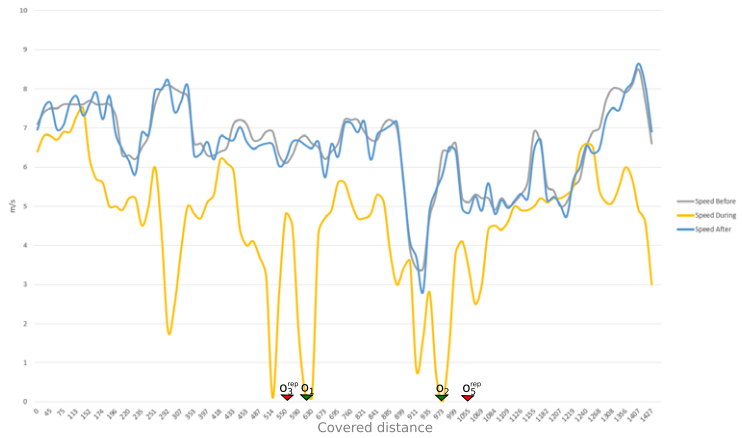


(e) Segment s_3: Evolution of the time marks

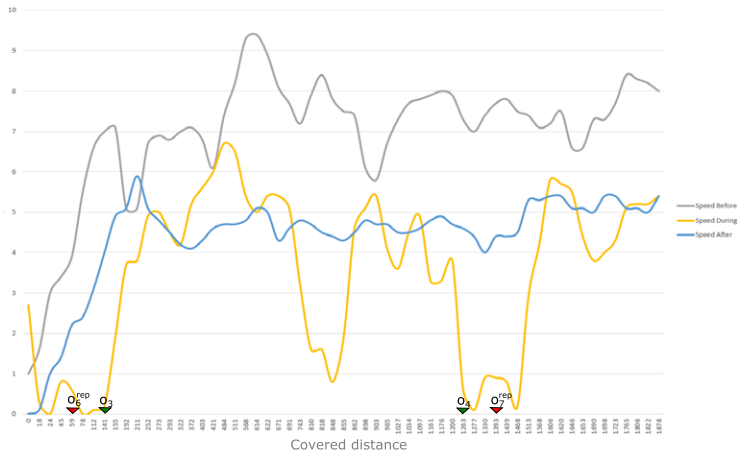


(f) Segment s_3: Distribution of the time marks per time period.

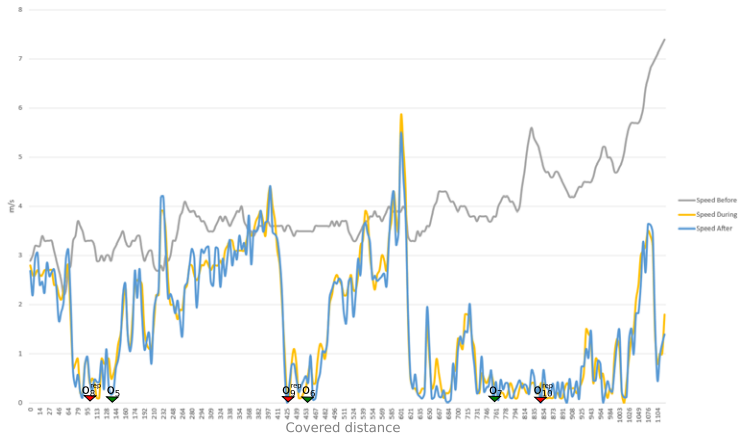
Fig. 9. Time marks of the cyclists in segments s_1, s_2 and s_3. Marks are labelled in three time periods based on the presence of the true obstacles and the repair.



(a) Segment s_1



(b) Segment s_2



(c) Segment s_3

Fig. 10. Average speed evolution of the trajectories for segments s_1, s_2 and s_3 before, during and after the appearance of the true obstacles. The red triangles represent the location of the inferred obstacles by SAMARITAN whereas the green ones indicates the true obstacles.

from the trajectories moving along these segments. This allows to detect obstacles o_1 and o_2 in segment s_1 (Fig. 10a) and obstacles o_2 and o_3 in s_2 (Fig. 10b) with very high accuracy. A similar behavior is observed in segment s_3 with obstacles o_5 and o_6 (Fig. 10c).

However, obstacle o_7 is located in an part of segment s_3 where the forest path was at a very bad condition. This caused all the trajectories in the last part of s_3 to move, on average, quite slowly for around 400 meters. As a result, many different ISLSPs were generated in that last part of the segment. This caused a slight displacement of the SMARITAN alert (o_{10}^{rep}) with respect the true obstacle.

Finally, SAMARITAN was not able to detect the true obstacle o_8 in an accurate manner (see Table 4). Since this obstacle is outside any segment it can only be detected by means of r-OSLSPs. According to Section 3.7, obstacles outside Strava segments are only generated when some close r-OSLSPs from the different cyclist are detected. From Fig. 11 we can see that only 2 out of 5 cyclist actually generated r-OSLSPs near o_8. This is because this obstacle is located in an area with a quite low transit of cyclists. Therefore, the time window $\mathcal{W}_{\text{r-OSLSP}}$, in charge of gathering the most recent r-OSLSPs, does not contain enough points so that the DBSCAN instance is able to uncover meaningful clusters.

4.4. Lessons learnt

From this use case we can draw up some interesting findings.

First of all, SAMARITAN uses two types of crowd-sensing data to detect forest-path problems, an internal and an external one. The former is the GPS trajectories generated by the cyclists acting as contributors that are explicitly processed by the client side of the framework. The latter is the road-segment statistics gathered from a third-party community service like Strava. In that sense, the use case has shown that the enrichment of a MCS architecture with crowd-based data from an external platform is a promising approach to extend the usage of these architectures to new domains such as the detection of road problems in rural regions.

Fig. 11. Location of all the OSLSPs generated during the experiment around obstacle o_8. The true obstacle location is shown as a green point. The points with the same color (blue or orange) are the OSLSPs from a particular cyclist.

Secondly, the evaluation of the framework has proved that, as many MCS architectures, the reliability and accuracy of the proposed solution strongly depends on the density of contributors. In that sense, road problems occurring within Strava segments are more likely to be discovered. This is because these segments are defined by the Strava users in parts of the road network covered by a large number of cyclists. This limitation is a side effect of relying on Strava as an input source.

However, this dependence allows SAMARITAN to leverage the data shared by users in Strava. As a result, it avoids a cold-start problem when it comes to retrieve historic speed-behaviour of cyclists of the segments in each new deployment. In that sense, if the Strava platform disappeared or restricted its access via API then it would be necessary to find an alternative public feed providing historic data about cyclist mobility in the target areas. In that sense, some spatial repositories, like OpenStreetMap, allows users to upload their own GPS trajectories.[6] Another possible alternative would be the composition of the segment statistics directly by SAMARITAN. This could be done in an incremental manner as long as the system processes data from the contributors. However, this option would suffer from the same cold-start problem described above.

Finally, the crowd-based approach followed by SAMARITAN to collect fitness-related data limits its application to scenarios accomplishing certain requirements. Since the system relies on sudden and abrupt deceleration of the cyclists as initial step to fire the path-problem detection mechanism, SAMARITAN would not be a feasible solution in urban environments where cyclists usually face many different *obstacles* (e.g traffic lights, pedestrians and so forth) that may make them to abruptly stop. This would generate plenty of noisy data and the system would report a large number of false positives.

[6] https://www.openstreetmap.org/traces

5. Conclusions

The endless enrichment of personal mobile contrivances with new sensing capabilities has enabled the development of many collaborative applications. In this context, MCS-based RIASs allow to monitor the state of large portions of a road network in a cost-effective manner. However, existing solutions focus on a rather limited scope as they assume that devices providing sensor data are used by car drivers.

In this context, the present work introduces SAMARITAN, a MCS-based RIAS that targets forest paths. This type of roads are very popular to do many different sports like cycling or hiking. However, they can also suffer from problems that affect their usual transit flow. These problems are caused by the presence of obstacles like fallen trees or landslides.

SAMARITAN profits from the growing popularity of fitness Internet applications to perform its path-monitoring task. In particular, it makes use of cyclist trajectories and community-based segments collected from Strava, a foremost sport application. The evaluation of the framework in a real-world scenario has shown that our solution has been able to accurately detect most of the target obstacles. However, as most solutions based on the MCS paradigm, its reliability depends on the density of contributors in the target area.

The key benefit of SAMARITAN with respect to other possible alternatives for forest-path monitoring is that it follows an opportunistic MCS approach. As a result, contributors do not need to explicitly notify the locations of the forest-path incidents because the system is able to do that based on their automatically-generated GPS trajectories. In that sense, potential alternatives where users must explicitly report the problems that they see in the paths would suffer from large usability problems. The most important one is that this type of approach would require users to stop halfway through their sport activity every time they spot an incident, report the problem and lastly resume their training. This would discourage the usage of these type of alternatives by many different potential contributors within the fitness field.

All in all, the present work would help rural administrations to better maintain their forest paths networks. It would allow these authorities to control large geographical areas at affordable cost provided that they are visited by enough cyclists.

Finally, future work will extend the framework to consider other contextual factors for obstacles detection. This way, the current weather conditions or the orography of the target region under study might be relevant features in order to asses whether an abnormal stop is caused by a road-condition problem or not.

Acknowledgements

Authors would like to thank the cyclists who kindly collaborated in the evaluation of the tool by giving up their GPS trajectories. Moreover, this work has been supported by the Fundación Séneca del Centro de Coordinación de la Investigación de la Región de Murcia under Project 20813/PI/18, and by the Spanish Ministry of Science, Innovation and Universities under grant RTC-2017-6389-5.

References

[1] G. Alessandroni, L. Klopfenstein, S. Delpriori, M. Dromedari, G. Luchetti, B. Paolini, A. Seraghiti, E. Lattanzi, V. Freschi, A. Carini et al., Smartroadsense: Collaborative road surface condition monitoring, in: *Proceedings of the UBICOMM*, 2014, pp. 210–215.

[2] I. Ayala, L. Mandow, M. Amor and L. Fuentes, A mobile and interactive multiobjective urban tourist route planning system, *Journal of Ambient Intelligence and Smart Environments* **9**(1) (2017), 129–144. doi:10.3233/AIS-160413.

[3] S. Basudan, X. Lin and K. Sankaranarayanan, A privacy-preserving vehicular crowdsensing-based road surface condition monitoring system using fog computing, *IEEE Internet of Things Journal* **4**(3) (2017), 772–782. doi:10.1109/JIOT.2017.2666783.

[4] L. Bayındır, A survey of people-centric sensing studies utilizing mobile phone sensors, *Journal of Ambient Intelligence and Smart Environments* **9**(4) (2017), 421–448. doi:10.3233/AIS-170446.

[5] C. Borcea, M. Talasila and R. Curtmola, *Mobile Crowdsensing*, CRC Press, 2016.

[6] C. Chen, Y. Ding, X. Xie, S. Zhang, Z. Wang and L. Feng, TrajCompressor: An online map-matching-based trajectory compression framework leveraging vehicle heading direction and change, *IEEE Transactions on Intelligent Transportation Systems* **21**(5) (2020), 2012–2028. doi:10.1109/TITS.2019.2910591.

[7] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden and H. Balakrishnan, The pothole patrol: Using a mobile sensor network for road surface monitoring, in: *Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services, MobiSys'08*, Association for Computing Machinery, New York, NY, USA, 2008, pp. 29–39. ISBN 9781605581392. doi:10.1145/1378600.1378605.

[8] M. Ester, A density-based algorithm for discovering clusters in large spatial databases with noise, in: *Proc. 2nd Int. Conf. on Knowledge Discovery and Data Mining*, AAAI Press, Portland, Oregon, 1996.

[9] G. Griffin, K. Nordback, T. Götschi, E. Stolz and S. Kothuri, Transportation research circular E-C183: Monitoring bicyclist and pedestrian travel and behavior: Current research and practice, Transportation Research Board of the National Academies, Washington, DC, 2014.

[10] G.P. Griffin and J. Jiao, Where does bicycling for health happen? Analysing volunteered geographic information through place and plexus, *Journal of Transport & Health* **2**(2) (2015), 238–247, http://www.sciencedirect.com/science/article/pii/S2214140514001042. doi:10.1016/j.jth.2014.12.001.

[11] F.E. Grubbs, Procedures for detecting outlying observations in samples, *Technometrics* **11**(1) (1969), 1–21. doi:10.1080/00401706.1969.10490657.

[12] H.H. Hochmair, E. Bardin and A. Ahmouda, Estimating bicycle trip volume for Miami-Dade county from Strava tracking data, *Journal of Transport Geography* **75** (2019), 58–69, http://www.sciencedirect.com/science/article/pii/S0966692318308639. doi:10.1016/j.jtrangeo.2019.01.013.

[13] J. Hong, D.P. McArthur and J.L. Stewart, Can providing safe cycling infrastructure encourage people to cycle more when it rains? The use of crowdsourced cycling data (Strava), *Transportation Research Part A: Policy and Practice* **133** (2020), 109–121.

[14] V. Janko, B. Cvetković, A. Gradišek, M. Luštrek, B. Štrumbelj and T. Kajtna, e-Gibalec: Mobile application to monitor and encourage physical activity in schoolchildren, *Journal of Ambient Intelligence and Smart Environments* **9**(5) (2017), 595–609. doi:10.3233/AIS-170453.

[15] P. Jonsson, J. Casselgren and B. Thörnberg, Road surface status classification using spectral analysis of NIR camera images, *IEEE Sensors Journal* **15**(3) (2015), 1641–1656. doi:10.1109/JSEN.2014.2364854.

[16] F. Kalim, J.P. Jeong and M.U. Ilyas, CRATER: A crowd sensing application to estimate road conditions, *IEEE Access* **4** (2016), 8317–8326. doi:10.1109/ACCESS.2016.2607719.

[17] L.C. Klopfenstein, S. Delpriori, P. Polidori, A. Sergiacomi, M. Marcozzi, D. Boardman, P. Parfitt and A. Bogliolo, Mobile crowdsensing for road sustainability: Exploitability of publicly-sourced data, *International Review of Applied Economics* (2019), 1–22. doi:10.1080/02692171.2019.1646223.

[18] A. Kulkarni, N. Mhalgi, S. Gurnani and N. Giri, Pothole detection system using machine learning on Android, *International Journal of Emerging Technology and Advanced Engineering* **4**(7) (2014), 360–364.

[19] X. Li and D.W. Goldberg, Toward a mobile crowdsensing system for road surface assessment, *Computers, Environment and Urban Systems* **69** (2018), 51–62, http://www.sciencedirect.com/science/article/pii/S0198971517301333. doi:10.1016/j.compenvurbsys.2017.12.005.

[20] L.C. Lima, V.J.P. Amorim, I.M. Pereira, F.N. Ribeiro and R.A.R. Oliveira, Using crowdsourcing techniques and mobile devices for asphaltic pavement quality recognition, in: *2016 VI Brazilian Symposium on Computing Systems Engineering (SBESC)*, 2016, pp. 144–149. ISSN 2324-7894. doi:10.1109/SBESC.2016.029.

[21] R. Mohamed, H. Aly and M. Youssef, Accurate real-time map matching for challenging environments, *IEEE Transactions on Intelligent Transportation Systems* **18**(4) (2017), 847–857. doi:10.1109/TITS.2016.2591958.

[22] W. Musakwa and K.M. Selala, Mapping cycling patterns and trends using Strava Metro data in the city of Johannesburg, South Africa, *Data in Brief* **9** (2016), 898–905, http://www.sciencedirect.com/science/article/pii/S235234091630662X. doi:10.1016/j.dib.2016.11.002.

[23] F. Palumbo, C. Gallicchio, R. Pucci and A. Micheli, Human activity recognition using multisensor data fusion based on reservoir computing, *Journal of Ambient Intelligence and Smart Environments* **8**(2) (2016), 87–107. doi:10.3233/AIS-160372.

[24] E. Rampinini, G. Alberti, M. Fiorenza, M. Riggio, R. Sassi, T. Borges and A. Coutts, Accuracy of GPS devices for measuring high-intensity running in field-based team sports, *International Journal of Sports Medicine* **36**(91) (2015), 49–53.

[25] G. Romanillos, M.Z. Austwick, D. Ettema and J.D. Kruijf, Big data and cycling, *Transport Reviews* **36**(1) (2016), 114–133. doi:10.1080/01441647.2015.1084067.

[26] B. Rosner, Percentage points for a generalized ESD many-outlier procedure, *Technometrics* **25**(2) (1983), 165–172. doi:10.1080/00401706.1983.10487848.

[27] R. Sileryte, P. Nourian and S. van der Spek, Modelling spatial patterns of outdoor physical activities using mobile sports tracking application data, in: *Geospatial Data in a Changing World*, T. Sarjakoski, M.Y. Santos and L.T. Sarjakoski, eds, Springer International Publishing, Cham, 2016, pp. 179–197. ISBN 978-3-319-33783-8. doi:10.1007/978-3-319-33783-8_11.

[28] G. Singh, D. Bansal, S. Sofat and N. Aggarwal, Smart patrolling: An efficient road surface monitoring using smartphone sensors and crowdsourcing, *Pervasive and Mobile Computing* **40** (2017), 71–88, http://www.sciencedirect.com/science/article/pii/S1574119216301262. doi:10.1016/j.pmcj.2017.06.002.

[29] Y. Sun and A. Mobasheri, Utilizing crowdsourced data for studies of cycling and air pollution exposure: A case study using Strava data, *International Journal of Environmental Research and Public Health* **14**(3) (2017), https://www.mdpi.com/1660-4601/14/3/274. doi:10.3390/ijerph14030274.

[30] Team AMS, Understanding GPS tracking, Technical report, Australia, 2013.

[31] X. Wang, J. Zhang, X. Tian, X. Gan, Y. Guan and X. Wang, Crowdsensing-based consensus incident report for road traffic acquisition, *IEEE Transactions on Intelligent Transportation Systems* **19**(8) (2018), 2536–2547. doi:10.1109/TITS.2017.2750169.

[32] Z. Yan and D. Chakraborty, Semantics in mobile sensing, *Synthesis Lectures on the Semantic Web: Theory and Technology* **4**(1) (2014), 1–143. doi:10.2200/S00577ED1V01Y201404WBE008.

[33] J. Zhang and M.F. Goodchild, *Uncertainty in Geographical Information*, CRC Press, 2002.

[34] Y. Zheng, Trajectory data mining: An overview, *ACM Trans. Intell. Syst. Technol.* **6**(3) (2015). doi:10.1145/2743025.