Efficient Algorithms for Accuracy Improvement in Mobile Crowdsensing Vehicular Applications

Saverio Delpriori, Valerio Freschi, Emanuele Lattanzi and Alessandro Bogliolo

Department of Basic Sciences and Foundations
University of Urbino, Urbino, Italy
Email: saverio.delpriori@uniurb.it, valerio.freschi@uniurb.it, emanuele.lattanzi@uniurb.it, alessandro.bogliolo@uniurb.it

***Abstract*—Mobile crowdsensing is emerging as a cost-effective solution to conduct extensive monitoring campaigns by exploiting the potential of mobile terminals with unprecedented communication, processing, and sensing capabilities. On the other hand, data provided by a large number of end-users equipped with heterogeneous devices pose trust and accuracy issues that might impair the overall reliability and usability of the system. Trust management techniques developed in the field of online social networks can be effectively used to detect and isolate cheating users, but they cannot avoid the risk of inaccurate data provided by trustworthy agents because of the inherent limitations of their devices or of the adverse conditions in which they operate. This work presents efficient algorithms for compensating the inaccuracy of crowdsensing geospatial data to be reported on a road map. The paper illustrates on a representative case study the main issues of map matching and the effectiveness of the proposed solutions, which belong to the category of incremental online methods targeting dense sampling points to be mapped on road lines without topological annotations.**

Keywords–Map Matching; Crowdsensing; Vehicular Application.

I. INTRODUCTION

The widespread diffusion of smart mobile devices equipped with sensors and GPS receivers has enabled the development of crowdsensing applications engaging end-users in environmental/urban monitoring tasks. If, on one hand, crowdsensing has the potential of providing huge amount of data at negligible cost, it has some inherent limitations that need to be carefully addressed. First of all, end-users need to be motivated to take part in monitoring campaigns in order to make sure that a sufficient amount of data is provided. Second, misbehaviours (due either to unskillfulness or to cheating) need to be recognized and isolated in order not to affect the reliability. Third, data accuracy is not guaranteed because of the lack of control on the inherent accuracy of the devices and on their operating conditions.

The fast growing interest in crowdsensing has prompted for the development of a large number of approaches to address each one of the above mentioned issues, as documented in recent works surveying cooperation incentives [1], trust management [2], and crowdsensing data accuracy [3].

When dealing with geo-spatial data, accuracy is a two-fold issue, in that it concerns both the value of the sensed quantity and the GPS coordinates of the point in which it was measured. When the point needs to be associated with an object on a map, GPS accuracy impacts map matching [4], [5].

This paper focuses on efficient map matching algorithms for crowdsensing road applications. In particular, it provides a classification of map matching issues and proposes incremental real-time algorithms to tackle them and improve the overall mapping accuracy.

A mobile crowd-sensing application for road surface monitoring, called *Smart Road Sense* (SRS), is taken as a case study [6]. Among the wide range of vehicular applications that pose map matching issues, SRS has several peculiarities that give rise to new challenges: i) it requires fine-grained sampling, ii) it targets not only recognized main roads, but also not-yet-tagged road segments, and iii) it collects data from heterogeneous devices installed both on public and on private vehicles. Hence, map matching is addressed without relying on a pre-established trajectory (as in the case of public transportation), nor on the knowledge of roads and links among them. Rather, incoming data are sequences of points that need to be matched on road segments based only on geometrical considerations and on first-order reachability analysis performed on the fly. Moreover, we assume map matching to be applied in real time, so that computational complexity has to be kept under tight control, looking for a solution belonging to the category of incremental online methods.

Experimental results, validated against ground truth datasets collected for one month on known bus lines, show that map matching accuracy can be significantly improved by means of incremental linear algorithms applied on the fly without leveraging any topological information.

The rest of the paper is organized as follows: Section II provides a summary of related work on vehicular crowdsensing systems and map-matching algorithms; Section III introduces concepts and definitions related to the map-matching problem; Section IV describes the proposed approach; Section V describes the experimental setup and presents the results; Section VI concludes the work.

II. PREVIOUS WORK

In this section we report some recent scientific literature related to techniques, methods and systems used in the paper, namely: crowdsensing, system architectures for mobile sensing (with a particular focus on vehicular applications) and algorithmic approaches for map matching problems.

Crowdsensing denotes a whole class of applications executed by many individuals equipped with mobile devices featuring sensing, computational and communication capabilities. Modern smartphones are today commonly equipped with a wide range of sensors (e.g., GPS, accelerometers, microphones, cameras, etc.). This fact, combined to the growing diffusion of these devices and to the possibility of geo-

 ISBN: 978-1-61208-418-3

referencing collected data, makes them particularly suitable as an enabling technology for supporting mobile crowdsensing systems [7], [8].

A. Vehicular crowdsensing systems

One of the first applications of vehicle-based crowdsensing is represented by *CarTel*, a system developed with the aim of detecting road potholes by means of GPS and accelerometers mounted in cars equipped with embedded microprocessors [9]. Mobile sensing for the detection of traffic conditions, bumps and acoustic events has been investigated through the integration of data from accelerometers and microphones into a system called *Nericell* [10]. Thiagarajan *et al.* proposed in 2009 a system (named *VTrack*) targeting the goal of road traffic delays estimation by means of mobile phones [4]. A particular focus of this work is represented by the reduction of smartphone energy consumption during sensing activities and by the compensation of noise associated to sensors sampling. *CTrack* is a system developed by some of the authors of VTrack to achieve accurate trajectory mapping from GSM fingerprints instead of using WiFi signals or GPS traces [11].

Large-scale mobile sensing has been proposed for air pollution monitoring in *OpenSense* [12]. Data gathering in OpenSense is achieved by means of participative sensing of citizens equipped with enhanced modified smartphones or ad hoc pocket sensors, and by means of special sensor stations placed on public transport vehicles.

Recently, a system architecture for road surface collaborative monitoring (termed *SmartRoadSense*, SRS) has been proposed [6], [13]. SRS is designed to integrate mobile sensing and cloud systems to support continuous monitoring of road surface quality. A roughness index is computed on board of end-users' smartphones and then transmitted to be stored, processed, aggregated and visualized in cloud.

B. Map matching

Map matching is an inference process that reports a sequence of location data onto a map. In mobile sensing applications, data is typically a GPS trace (i.e., a sequence of time-stamped *(latitude, longitude)* values) and the map refers to annotated road networks in a digital georeferenced database. Sequence localization data can derive from different sources of information (e.g., GPS devices, GSM fingerprints, WiFi access points position) while target maps could differ in the information content and available details (e.g., topological information, one-way roads annotations, etc.).

Map-matching algorithms are classified into *global* algorithms and *incremental* algorithms. Global approaches [5], [11] process whole input traces in order to achieve a solution, while incremental algorithms [14], [15] work on small segments to be processed in sequential order. On one hand, global algorithms usually result into more accurate solutions but must be inherently run only offline; on the other hand, incremental algorithms make local choices that could possibly impact the accuracy of mapping but are suitable for online execution.

In this framework, many different techniques have been proposed to tackle the map-matching problem. Some authors investigated the use of computational geometry techniques posing the problem as a pattern matching between curves under Frechet distance [16], [17]. Others proposed the use of signal processing methods (e.g., Extended Kalman Filters) [18] or Bayesian estimators [15]. According to recent scientific literature [19], the best performances in terms of accuracy are obtained by algorithms based on *Hidden Markov Models* (HMMs). This statistical approach grants robustness to the matching algorithm, resulting into enhanced accuracy when faced with noisy inputs [5], [11]. A local, incremental variant amenable for an online implementation has been recently presented by Goh *et al.* [19].

Despite significant advancements obtained, the above mentioned algorithmic approaches present some issues with respect to the mobile crowdsensing scenario targeted by this paper. In fact, global solutions incur high computational overhead that make them unsuitable for vehicle-based applications where timeliness is often required. Furthermore, most state-of-the art methods make some assumptions on input data (e.g., the availability of topological information) which are not always guaranteed. Overcoming these issues is one of the main purposes of this work.

III. Problem statement and scope

This section formulates the map-matching problem addressed in this paper referring to known definitions [19] and to the terms adopted in OpenStreetMap [20].

Definition 1: A *trace*, $T = (t_n | n = 1, ..., N)$, is a sequence of N samples collected by a vehicle. Each *trace point* t_n is characterized at least by: time stamp (t_n.t), GPS coordinates (t_n.lat, t_n.lon), and sample (t_n.val). Additional fields can be available, like the GPS accuracy (t_n.acc) or the vehicle speed (t_n.v).

Definition 2: A *line*, $L = (p_m | m = 1, ..., M)$, is a M-point polyline representing a road segment as a series of segments connecting vertices p_1, ..., p_M in order. Each *vertex* p_m is represented by its coordinates (p_m.lat, p_m.lon).

Definition 3: A *map* S is a set of lines representing all the road segments of interest.

According to OpenStreetMap, we consider a *road* as a relation among lines, possibly annotated with viability information (speed limit, directions, permissions, etc.). A road network is a set of roads complemented by a connection matrix which adds topological information to the map. A road network is defined on a map and, in general, it covers just a subset of the lines of the map. The percentage of lines covered by the road network in a given region depends on the maintenance and updating of the underlying data base. Working on a map (rather than on a road network) maximizes the coverage at the cost of giving up topological information, which are usually exploited in map-matching algorithms.

Definition 4: Given a trace T and a map S, the goal of *map matching* is to find the correspondence between each point of T and a line of S.

In general, reducing the sampling rate (i.e., reducing the number and density of the points in the trace) makes it more difficult to determine the actual trajectory of the vehicle on the map. Hence, large research efforts have been devoted to the development of robust map matching solutions able to cope with sparse traces [4], [5]. In the limiting case in which there is less than 1 sample per line, the key problem is to figure out which path the vehicle took between two points. The topology of the road network is essential in this case.

 ISBN: 978-1-61208-418-3

On the other hand, challenging issues can be raised also by the abundance of points provided by high sampling rates. When the rate reaches the order of one sample per second, there are two new problems to face. The first one is accuracy, in that the distance between subsequent points in the trace might fall below the resolution of the GPS, causing many possible artifacts. The second one is performance, in that the sampling rate poses tight constraints on the time taken to process each sample in online real-time applications.

A. Map-matching issues

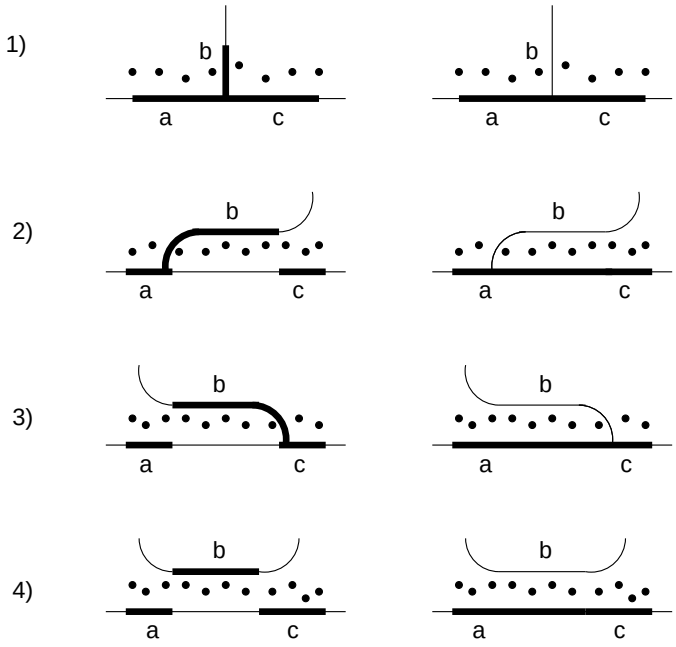

Figure 1. Schematic representation of the 4 main artifacts of map matching applied to dense traces. For each case the artifact is represented to the left and contrasted with the ground truth trajectory, to the right.

Figure 1 provides a schematic representation of the basic artifacts possibly produced by map matching algorithms fed by a dense trace. Lines (i.e., road segments) are denoted by labels a, b and c, while dots represent trace points. Bold lines on the left represent the false trajectory inferred by matching points on the closest lines, while those on the right represent the actual (ground-truth) trajectory. All the traces are assumed to go from left to right (meaning that points to the left have time stamps preceding those to the right) along a ground-truth trajectory that goes from a to c. Label b is used to denote a line which is not in the trajectory but passes so close to some of the trace points to induce mapping artifacts.

In $Case1$ line b is a crossing road orthogonal to a and c. The wrong trajectory looks as if the vehicle had taken the cross and then was immediately turned back on the main road.

In $Case2$ line b is one of the two roads of a fork which runs parallel to c for a non-negligible stretch. If, after the fork, most of the trace points are closer to b than to c, it looks as the fork was taken. But when b and c diverge, the remaining points are mapped back to c without a viable link.

In $Case3$ a fork similar to that of Case 2 is encountered in the opposite direction, so that the fake trajectory jumps from a to b as the two roads get close to each other, and then it rejoin the main road at the fork.

In $Case4$ some of the samples are mapped on line b even if it never encounters lines a and b.

All the artifacts possibly encountered when map-matching a dense trace can be expressed as a combination of the four listed above. Without loss of generality, in cases 1, 2, and 3 we assume the road to switch from line a to line b exactly when it crosses b. All other situations can be easily obtained by considering a=c. For the sake of explanation, the points in Figure 1 are represented as equidistant from each other and laying on an ideal trajectory, even if misplaced with respect to the map. Real traces look much worse, in that the points usually jump on both sides of the ground-truth lines and sometimes they overlap and locally reverse the order. However, persistent errors like the ones schematically represented in Figure 1 are the most difficult to detect and correct. It is also worth mentioning that, in the absence of viability or topological annotations, the artifacts of Figure 1 occur whenever there are lines that cross or get close to each other on the map, even if there are no paths between them in the road network.

The purpose of this work is to find an incremental map-matching algorithm able to detect and correct matching artifacts when dealing with dense traces in the absence of topological information.

IV. Proposed approach

The output of map matching is the association of each trace point to a line. We use t_n.line to denote the property of point t_n that represents its association to a given line. Hence, map matching reduces to setting all the t_n.line properties to the appropriate line id's.

Restricting the range of candidate solutions to the ones that can be executed incrementally in linear time, we start by matching each point to the closest line, as determined by issuing a query to the underlying geospatial data base. To speed up the refinement of this first-cut matching we make use of *run length encoding* (RLE) to represent contiguous subsets of trace points mapped on the same line [21].

Definition 5: A *run* $R = (r_k|k = 1, ..., K)$, is a sequence of K contiguous points taken from a given trace, such that all the points in R are mapped on the same line (denoted by R.line) and the points that precede and follow R in the trace (if any) are mapped on a different line.

Runs are computed on the fly based only on proximity matching, and then incrementally processed to correct artifacts. A sliding window of three runs, denoted by $previous$ (R_p), $current$ (R_c) and $next$ (R_n), is used to this purpose. At each step a decision is taken on the correctness of the matching of R_c, based on the consolidated matching of R_p and on the first-cut matching of R_n.

The decision process is based on the following conditions, which minimize the number of additional queries:

A) Reachability from previous run. R_c.line and R_p.line cross in a point close to the first point of R_c and to the last of R_p (only two points and two lines involved in the query);

B) Reachability of next run. R_n.line and R_c.line cross in a point close to the first point of R_n and to the last of R_c (only two points and two lines involved in the query);

C) Fork. R_p.line, R_c.line and R_n.line cross in the same point, close to the first and last points of R_c (only two points and three lines involved in the query);

D) Short run. The number of points in R_c is lower than a given significance length L (no geospatial queries required).

TABLE I. TRUTH TABLE OF THE ARTIFACT COMPENSATION ALGORITHM.

	A	B	C	D	Action
Case 1	T	T	T	T	split R_c between R_p and R_n
Case 2	T	F			merge R_c in R_p
Case 3	F	T			merge R_c in R_n
Case 4	F	F			split R_c between R_p and R_n

Table I shows the decision criteria used to detect the 4 cases of Figure 1 and the actions taken to correct each of them. Rows are associated with possible artifacts, while columns are associated with conditions. Empy entries represent don't care conditions.

In Cases 1 and 4, the current run (R_c) is split between the previous and the last ones. This is done by assigning R_p.line to the line property of the first points of R_c, and R_n.line to the remaining points of R_c. In Cases 2 or 3, the current run is merged enither in the previous one or in the next one.

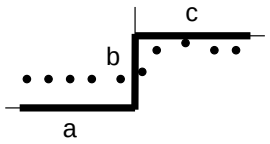

Figure 2. False positive case: actual trajectory that risks to be recognized as a case-1 artifact if the link road *b* is very short.

Looking only at the truth table, conditions C and D seem to be redundant. In fact, the two binary conditions A and B provide the 4 combinations required to encode the 4 cases of interest. However, C and D are needed to distinguish Case 1 from fals positives, like the one in Figure 2, which represents a short link road. Whenever none of the four artifacts is detected, the current run is assumed to be correct and its line labels are left unchanged.

V. IMPLEMENTATION AND EXPERIMENTAL RESULTS

This section presents the experimental results obtained by applying the proposed map matching algorithm to enhance the quality of the SRS road surface monitoring system.

A. Implementation and experimental setup

The architecture of SRS is composed of three components: a mobile application running on Android devices to compute every second a geo tagged estimate of a roughness index, a server that gathers roughness traces from all the end-user devices and map them on OpenStreetMap, and a cloud-based front-end for graphical visualization of geospatial data, based on CartoDB.

The server-side map matcher of SRS, called *Cartesian matcher*, associates each trace point to the closest line, according to the 2-dimensional Cartesian distance. The proposed algorithm is implemented as an additional component, called *match enhancer*, which operates on the runs of trace points annotated by the Cartesian matcher.

Data points are collected by a standard web service endpoint and stored in a PostGIS database system. Both the Cartesian matcher and match enhancer are implemented as PHP scripts, which operate on collected data by means of PostGIS stored procedures written in PL/pgSQL. Maps are taken from the OpenStreetMap project, while CartoDB is used to display data and double-check the correctness of the matches.

Although SRS was conceived as a crowdsensing application engaging end-users in road surface monitoring, the approach was validated on data collected for one month from Android devices installed on public buses operating on a known line in the Province of Pesaro-Urbino, in Italy. All the map elements along the actual bus line were manually sorted to build the ground truth baseline.

B. Results

Figures from 3 to 7 show the results of map matching in the four misleading cases targeted by the proposed approach. Each figure refers to a specific case and reports the trace points projected on the line segments in which they are mapped by the traditional Cartesian matching and by the proposed match enhancer. The effectiveness of the enhancer is apparent, in that the artifacts are completely cancelled. In particular, Figure 3 shows several examples of Case-1 artifacts, Figure 4 shows an example of Case 2, Figure 5 shows an example of Case 3, while both Figure 6 and Figure 7 provide examples of Case 4.

Two metrics can be used to estimate the accuracy of map matching relative to ground truth: the percentage of trace points mapped on lines that belong to the bus line (i.e., the correct segment identification rate [22]), and the ratio between the number of wrong lines and the number of ground-truth lines involved in matching (fake line ratio). Cartesian matching provided a correct segment identification rate of 97.15%, but in spite of the small percentage of wrong matches, the fake lines involved in matching exceeded the number of ground-truth lines (108 against 102), leading to a ratio of 1.06. The difference between the two metrics is explained by the nature of the trace and of the map: the trace includes extra urban roads with a limited occurrence of artifacts, while the map is highly fragmented, containing many unclassified very short segments, which are the most error prone ones.

The application of match enhancement provided sizeable advantages, increasing the correct segment identification rate to 99.10%, and the fake line ratio to 0.47.

VI. CONCLUSIONS

This paper has presented an incremental real-time algorithm to enhance the accuracy of map matching in crowdsensing applications dealing with dense traces to be mapped on maps with unknown topology. The issues raised by the abundance of trace points have been discussed and classified and efficient solutions have been proposed to handle them on the fly. The effectiveness of the match enhancing techniques

 ISBN: 978-1-61208-418-3

(a) (b)

Figure 3. Projected trace points plotted on top of the satellite map showing examples of Case-1: (a) before and (b) after match enhancement.

(a) (b)

Figure 4. Projected trace points plotted on top of the satellite map showing examples of Case-2: (a) before and (b) after match enhancement.

(a) (b)

Figure 5. Projected trace points plotted on top of the satellite map showing examples of Case-3: (a) before and (b) after match enhancement.

presented in this paper has been demonstrated on a real-world case study: a collaborative system for road surface monitoring.

Experimental validation performed on known trajectories shows that all types of artifacts are properly handled by the proposed algorithm with significant improvements of the overall matching accuracy. In particular, the percentage of correct matches increases from 97.15% to 99.10%, while the ratio between fake lines and correct ones reduces from 1.06 to 0.47.

REFERENCES

[1] L. Duan et al., "Incentive mechanisms for smartphone collaboration in data acquisition and distributed computing," in IEEE Proceedings INFOCOM, 2012. IEEE, 2012, pp. 1701–1709.

[2] T. French, E. Asimakopoulou, C. Maple, and N. Bessis, "Trust Issues on Crowd-Sourcing Methods for Urban Environmental Monitoring," Int. J. Distrib. Syst. Technol., vol. 3, no. 1, 2012, pp. 35–47.

[3] S. Sarma, N. Venkatasubramanian, and N. Dutt, "Sense-making from Distributed and Mobile Sensing Data: A Middleware Perspective," in Proc. of the 51st Annual Design Automation Conference, ser. DAC '14. ACM, 2014, pp. 68:1–68:6.

[4] A. Thiagarajan et al., "VTrack: Accurate, Energy-aware Road Traffic Delay Estimation Using Mobile Phones," in Proc. of the 7th ACM Conference on Embedded Networked Sensor Systems. ACM, 2009, pp. 85–98.

 ISBN: 978-1-61208-418-3

(a) (b)

Figure 6. Projected trace points plotted on top of the satellite map showing examples of Case-4: (a) before and (b) after match enhancement.

(a) (b)

Figure 7. Projected trace points plotted on top of the satellite map showing examples of Case-4: (a) before and (b) after match enhancement.

[5] P. Newson and J. Krumm, "Hidden Markov Map Matching Through Noise and Sparseness," in Proc. of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ser. GIS '09. ACM, 2009, pp. 336–343.

[6] G. Alessandroni et al., "SmartRoadSense: Collaborative Road Surface Condition Monitoring," in Proc. of 8th International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, Ubicomm 2014. IARIA, 2014, pp. 210–215.

[7] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," Communications Magazine, IEEE, vol. 49, no. 11, 2011, pp. 32–39.

[8] N. D. Lane et al., "A survey of mobile phone sensing," Communications Magazine, IEEE, vol. 48, no. 9, 2010, pp. 140–150.

[9] 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 Proc. of the 6th international conference on Mobile systems, applications, and services. ACM, 2008, pp. 29–39.

[10] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: Rich Monitoring of Road and Traffic Conditions using Mobile Smartphones," in Proc. of the 6th ACM conference on Embedded network sensor systems. ACM, 2008, pp. 323–336.

[11] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod, "Accurate, Low-energy Trajectory Mapping for Mobile Devices," in Proc. of the 8th USENIX Conference on Networked Systems Design and Implementation, ser. NSDI'11. USENIX Association, 2011, pp. 267–280.

[12] K. Aberer et al., "Opensense: open community driven sensing of environment," in Proc. of the 1st International Workshop on GeoStreaming (IWGS '10), 2010, pp. 39–42.

[13] V. Freschi et al., "Geospatial Data Aggregation and Reduction in Vehicular Sensing Applications: the Case of Road Surface Monitoring," in Proc. of 9th IEEE International Conference on Connected Vehicles, ICCVE 2014. IEEE, 2014.

[14] N. R. Velaga, M. A. Quddus, and A. L. Bristow, "Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems," Transportation Research Part C: Emerging Technologies, vol. 17, no. 6, 2009, pp. 672–683.

[15] O. Mazhelis, "Using recursive Bayesian estimation for matching GPS measurements to imperfect road network data," in 13th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2010, Sept 2010, pp. 1492–1497.

[16] S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk, "On map-matching vehicle tracking data," in Proc. of the 31st international conference on Very large data bases. VLDB Endowment, 2005, pp. 853–864.

[17] D. Chen, A. Driemel, L. Guibas, A. Nguyen, and C. Wenk, "Approximate Map Matching with respect to the Frechet Distance," in Proc. of the 13th Workshop on Algorithm Engineering and Experiments, ALENEX'11. SIAM, 2011, pp. 75–83.

[18] D. Obradovic, H. Lenz, and M. Schupfner, "Fusion of Map and Sensor Data in a Modern Car Navigation System," Journal of VLSI signal processing systems for signal, image and video technology, vol. 45, no. 1-2, 2006, pp. 111–122.

[19] C. Goh et al., "Online map-matching based on Hidden Markov model for real-time traffic sensing applications," in 15th International IEEE Conference on Intelligent Transportation Systems (ITSC), 2012, Sept 2012, pp. 776–781.

[20] "OpenStreetMap," 2014, URL: http://www.openstreetmap.org [accessed: 2015-01-02].

[21] D. Salomon, Data compression: the complete reference. Springer, 2004.

[22] M. Hashemi and H. A. Karimi, "A critical review of real-time map-matching algorithms: Current issues and future directions," Computers, Environment and Urban Systems, vol. 48, 2014, pp. 153–165.

 ISBN: 978-1-61208-418-3