Detection and Localization of Traffic Signals with GPS Floating Car Data and Random Forest

Yann Méneroux
Univ. Paris-Est, LASTIG COGIT, IGN, ENSG, Saint-Mandé, France
yann.meneroux@ign.fr

Hiroshi Kanasugi
CSIS, Institute of Industrial Sciences, The University of Tokyo, Japan

Guillaume Saint Pierre
Centre for Studies and Expertise on Risks, Mobility, Land Planning and the Environment (Cerema), Toulouse, France

Arnaud Le Guilcher
Univ. Paris-Est, LASTIG COGIT, IGN, ENSG, Saint-Mandé, France

Sébastien Mustière
Univ. Paris-Est, LASTIG COGIT, IGN, ENSG, Saint-Mandé, France

Ryosuke Shibasaki
CSIS, Institute of Industrial Sciences, The University of Tokyo, Japan

Yugo Kato
Transport Consulting Division, NAVITIME JAPAN Co., Ltd

Abstract

As Floating Car Data are becoming increasingly available, in recent years many research works focused on leveraging them to infer road map geometry, topology and attributes. In this paper, we present an algorithm, relying on supervised learning to detect and localize traffic signals based on the spatial distribution of vehicle stop points. Our main contribution is to provide a single framework to address both problems. The proposed method has been experimented with a one-month dataset of real-world GPS traces, collected on the road network of Mitaka (Japan). The results show that this method provides accurate results in terms of localization and performs advantageously compared to the *OpenStreetMap* database in exhaustivity. Among many potential applications, the output predictions may be used as a prior map and/or combined with other sources of data to guide autonomous vehicles.

2012 ACM Subject Classification Computing methodologies → Machine learning, Information systems → Global positioning systems, Information systems → Data mining

Keywords and phrases Map Inference, Machine Learning, GPS Traces, Traffic Signal

Digital Object Identifier 10.4230/LIPIcs.GIScience.2018.11

Acknowledgements We would like to express our gratitude to Paul Chapron, Prof. Harutoshi Yamada and Prof. Yukio Sadahiro for their precious advice and constructive criticism.

1 Introduction

As one of the main supports for citizen mobility, roads are deservedly considered as a major cartographic theme in maps. Therefore, it is not surprising that most national mapping agencies allocate considerable amount of resources to keep road network databases as detailed,

10th International Conference on Geographic Information Science (GIScience 2018).
Editors: Stephan Winter, Amy Griffin, and Monika Sester; Article No. 11; pp. 11:1–11:15
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

accurate and up-to-date as possible [14, 4]. This is generally done by stereorestitution on aerial orthoimages [17], completed with field surveys to get details that cannot be captured in the images. Recently, automatic detection of roads has dramatically improved, especially when combined with machine learning algorithms [28], and now achieves very good performance even on satellite images. However, if the whole process tends to get less expensive and less time-consuming, it still suffers from a major drawback: road map timeliness is inevitably limited by the frequency of aerial image release [7].

Nowadays, with the spread of connected terminal devices equipped with a Global Positioning System (GPS) receiver, an increasing number of vehicle trajectories are becoming available. *Map inference*, which aims at leveraging this new source of data to extract geographic information [3], is becoming popular and tends to complement, if not completely replace, traditional survey techniques. Among their main assets, GPS traces are recorded on a daily basis, which allows for short-delay update capabilities. Indeed, aerial picture campaigns are typically conducted every several years, notwithstanding an additional delay for image preprocessing and orthorectification. This substantial delay might be critical in applications relying on highly up-to-date reference networks, such as emergency routing or disaster mitigation.

Contrarily, with GPS traces analysis, modifications are potentially detectable as soon as enough traces are recorded on a suspicious point to ensure the statistical robustness of the notification. Ultimately, with connected devices, it is foreseeable that data will be recorded and processed by online algorithms, resulting in a much more reactive system that is capable of detecting ephemeral events (e.g., road works, detour or accidents) in quasi real-time. Moreover, data can be continuously recorded while drivers are commuting for example, which makes this solution much less expensive than aerial campaigns and field surveys. More anecdotally, since we may assume that for any consistent algorithm, the estimation is getting closer to the reality as the number of traces increases, the dataset sampling itself serves a logic of public utility: the most important itineraries are the most traveled, therefore those where road map inference is the most reliable.

Initially restricted to the construction of road geometry and topology, map inference is now getting attention for enriching pre-existing networks with attributes (number of lanes, speed limitations...) or infrastructure (traffic signals, speed bumps, bus stops...) [24, 18]. Most of these features are not accessible through aerial images, and utilizing GPS traces seems unavoidable. Moreover, aerial images may not be accessible in developing countries, or available only at prohibitive cost. Instead, access to data stemming from local fleets or collaborative transport smartphone applications, are producing large sets of GPS traces. This surrogate source of data may be used with map inference techniques to provide a cheap alternative solution for map construction.

An exhaustive and detailed knowledge of road infrastructure is a prerequisite to many applications. For example, autonomous cars are expected to appear on the market in the near future. Reliability and robustness of the information used by such vehicles to make decisions is a big concern. It is usually more reliable to know in advance the location and the type of object that should be detected and confirm detection with embedded sensors. Additionally, driving-assistance devices conception, road safety, eco-driving, urban traffic flow simulation or even accurate routing time computation are as many other examples of fields or applications where the knowledge of a road network needs to be completed with attributes and infrastructure [4, 26, 1].

In parallel, machine learning techniques are becoming all-pervasive in fields requiring to process a large amount of data, or simply when theoretical background is insufficient to build reliable predictive models. With this kind of approach, expert knowledge is no longer required, and algorithms are trained on labeled data. However, machine learning is a relevant solution only if the two following conditions are met: firstly we must have an extensive and representative training dataset, and secondly, we must have a natural definition of cost that quantifies how close the generated road map is compared to the training data ground truth. A few years ago, some authors such as Liu *et al.* [14], have introduced numerical measures to assess the quality of maps produced by GPS traces, hence opening the way for a full machine learning resolution of the problem [3]. In this vein, Zhang and Sester [27] combined fuzzy logic and k-means clustering for incrementally inferring maps, while Fathi and Krumm [10] proposed to train an Adaboost classifier to recognize road intersections, based on the probability density function of trace headings. Similarly, Van Winden *et al.* [25] found that Support Vector Machines (SVM) and regression trees are the most adequate algorithms for speed limit inference. In some more sophisticated algorithms, traces are combined with external sources of data to get better results, for example in [12] where Twitter data and SVM are used for an automatic mining of street names. We believe that statistical learning is especially adapted to this problem, and that it guarantees the portability of the approach to other cities, countries and environments.

Among traffic control devices, traffic signals are unarguably the most effective to regulate jammed intersections [23]. They have a crucial impact both on traffic flow at the city scale and on the perceptions of individual drivers. Surprisingly, very few research works address the problem of utilizing a collection of probe vehicle traces coupled with machine learning algorithms to detect traffic signals. The most related research work is certainly the one of M. Munoz-Organero *et al.* [19], who used machine learning algorithms to detect in real-time several kinds of road infrastructures, based on an analysis of speed and acceleration signals, estimated from GPS positions. Despite providing very good results, the performance scores clearly exhibits some limitation on traffic signal detection, compared to the cases of street crossings, and roundabouts. Besides when the only source of measurement is a GPS receiver, speed-based analysis is only possible provided that the GPS positioning is accurate enough (for example if equipped with a Doppler speed measurement, when used in differential mode, or in open areas) and sampling frequency is high (over 1 point per second or so). Furthermore, a natural extension of [19] would be to use all vehicles which traveled at a specific location to detect infrastructure. In this work, we propose a method to detect and then localize traffic signals through a random forest classification and regression using the spatial distribution of stop points along the road.

We must notice that localization is an important aspect of the problem. Even though we know that an intersection is controlled by a system of traffic lights, the positions of stop lines on each individual streets remain uncertain, and this is especially true since road network abstraction and generalization may introduce an additional component of uncertainty.

The remaining of the paper is structured as follows: the dataset and its preparation are briefly described in the next section, while our methodology to create instances, train and validate the model is detailed in section 3. Section 4 provides the results and discusses them. Eventually, section 5 concludes the paper.

2 Data and preparation

2.1 Study area

The experimentation was conducted in Mitaka (Japan), a commuter town located approximately 20 km west of central Tokyo, and covering an extent of 16 square kilometers. This choice was motivated by the fact that Mitaka contains a wide variety of urban aspects, ranging

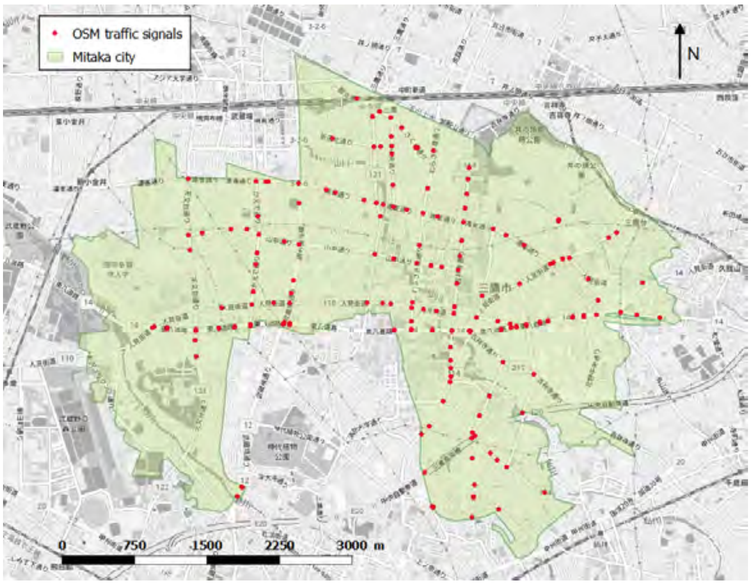

Figure 1 Mitaka city and OpenStreetMap traffic signal database.

from dense downtown to inter-urban residential districts, including motorway environments and parks as well. Mitaka city is illustrated on figure 1, where traffic signal controlled intersections are depicted in red.

We extracted a routable road map from the national reference. The topological graph of a road map is often organized in such a way that a node is always located close to each traffic signal, even when no physical intersection is involved (e.g., traffic signal associated to pedestrian crossing in the middle of a road link). For this reason, we decided to remove degree-2 nodes, so that it may practically be assumed that digital road network has been created without any knowledge of traffic signal locations.

2.2 Ground truth data acquisition

As an application of machine learning, it is necessary to collect ground truth data, namely the positions of all traffic signals in Mitaka, to train and then validate the algorithm. Throughout this paper, a *stop line* is defined as the position along the road, where the front vehicle in queue is expected to stop while waiting for the signal to turn green.

We started from a base reference extracted from *OpenStreetMap* (figure 1). This source of data is not complete, and each point corresponds to an entire crossing controlled by a system of traffic lights, but no information is provided regarding the number of streets actually controlled by an individual signal, nor are the positions of stop lines on these streets. Using OSM basis and multiple sources of orthoimages (produced at different dates), positions of stop lines have been manually digitized, and then orthogonally projected onto the road network, as depicted here after on figure 2. At the end of this process, a total of 669 stop line positions have been digitized, which corresponds to 253 crossings controlled by traffic signals. Out of them, 177 (70%) are reported in OSM database. For each stop line, we also recorded a binary attribute to indicate which direction of flow is subject to stop at the traffic signal. It takes the value 0 if the stop line is directed to vehicles traveling from source node to target node, otherwise it is set equal to 1 (source and target node is arbitrarily defined by the road network database provider).

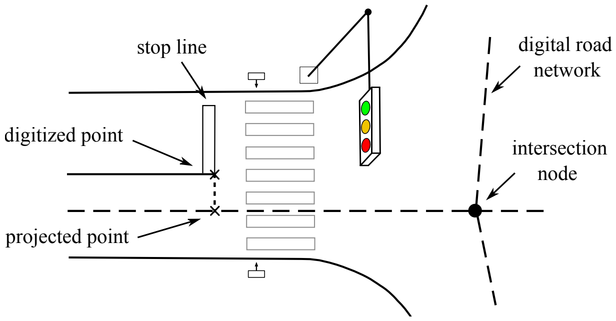

Figure 2 Ground truth data acquisition on orthoimages and reference road network.

Eventually, since orthoimages might suffer from local distortions, we had to check that our ground truth dataset is accurate enough for our application. A positional accuracy control was carried out by uniformly sampling 30 stop lines at random and surveying them with a single phase low-cost GPS receiver [16]. This operation enabled to guarantee (with 95% confidence index) that stop line positions have been digitized with a sub-meter accuracy (root mean square error below 90 cm).

2.3 Floating Car Data

For this experimentation, we used GPS Floating Car Data (FCD) provided by NAVITIME JAPAN[1], a private company developing navigation technologies and providing various kinds of web application services such as route navigation, travel guidance, and other useful information services for moving people.

The sample dataset is covering the entire Japan and has been recorded over a one-month time span, in October 2015. Pedestrian trajectories have been priorly removed so that it contains only vehicle navigation data. We extracted all GPS records intersecting the Mitaka polygon shape. Each record (nominally one per second) contains the following entries: a user identification number, a route identification number, geographic coordinates (in decimal degrees) and a timestamp. A route is a set of records on an individual subtrip (i.e. during a GPS receiver session). Due to privacy issues, driver identification number is modified every day at midnight. Entries containing -1 in timestamp or coordinates (i.e. about 2% of records, corresponding to GPS signal lost or logging failure) have been removed. Coordinates (as well as network and traffic signal ground truth) have been converted into UTM 54N cartographic projection system. For convenience purposes, we also transformed timestamps into an integer number of epochs. This made computing traveled distances and elapsed time between records much easier.

Similarly to most studies related to GPS probe vehicles, map-matching, which consists in reconstructing the path traveled by a vehicle on a network, is an important pre-processing step and has two, possibly combined, main advantages: providing a mapping function between GPS positions and network links (which is necessary in our application case for updating the network) and enhancing positional accuracy. The latter is particularly important in urban environment, where GPS satellite signal is likely to be partially impeded by buildings. We used an algorithm based on Hidden Markov Models, developed by Newson and Krumm [20].

[1] http://corporate.navitime.co.jp/en

Since all traces are located on the same area, it is worthwhile to compute shortest path distances between every couple of nodes just once, then storing results in a look-up table, before map-matching all trajectories in a batch. Following this approach enabled to speed-up the process, and reach a pace of 10 traces map-matched per second (approximately 1500 faster than the naive solution requiring to process shortest paths online). However, for a road network containing a number n of nodes, since the time and space complexities of the look-up table computation are growing like $\mathcal{O}(n^2)$, it inevitably becomes necessary to find alternative solutions when the area of interest is large. One of them might be to use sparse matrix notation with hashtable data structure, and save only distances which are shorter than a predefined threshold.

Root mean square error of displacements induced by map-matching is equal to 8.3 m, which gives some insight regarding the average quality of GPS receivers. Overall, 99% of records have been map-matched (excluding outlier points). Eventually, we removed all traces map-matched with Chūō expressway, which runs the south-eastern part of Mitaka and, needless to say, does not contain any traffic light.

At the end of the pre-processing phase, a total of 11870 traces are remaining in the dataset, which represents slightly above 7 million records, about 42000 km and 3122 hours of driving data. The median trip runs 3 km and lasts 10 minutes. 95% of the dataset is recorded at a frequency higher than 0.2 Hz.

3 Methodology

In this section, we describe our methodology to build training and validation instances from GPS trajectories, then after a brief review of Random Forest algorithm, we present an extension to aggregate individual predictions, and infer the presence of traffic signals at the level of crossings.

3.1 Instance computation

In most machine learning problems, there is a natural definition of an instance. For example, in image recognition tasks, each individual image is an instance, and we may easily assume that they are independent to each other. In our application case, there is no such definition, since we are looking for objects located at unknown positions on a topological network. However, considering that most traffic signals are located near intersections, we decided to compute instances based on road segments starting from nodes. This choice was motivated by the fact that it results in mutually independent instances, hence facilitating split process into training and validation datasets. In turn, our algorithm will inevitably fail to detect traffic signals located far from road intersections. Since, it may be assumed that this represents a small proportion of all traffic signals, we believe that this choice would not have too much negative impact. Note that, as depicted on figure 3, each network edge is generating two instances (one starting from each node). Hence the total number of instances generated equals at most twice the number of edges in the road network (in fact, some of them might be empty of traces, consequently the actual number of instances is generally smaller). We will refer to this segment as a *frame* hereafter.

In order to get homogenous instances, frames have been set to a fixed length L. If an edge is longer than L, then only a portion of length L (starting from the node) is considered. On the opposite, if it is shorter than L, the frame is padded with zeroes (X_5 and X_6 on figure 3). The numerical value of L was set to 100 m, since there is no evidence to think that events located further than 100 m from a traffic signal, might be of any help for the detection.

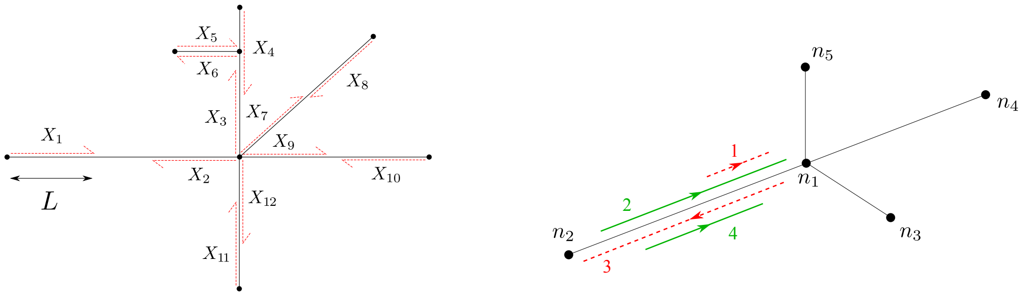

Figure 3 Left: frames generation (red dashed arrows) on the road network. Each frame is computed based on GPS traces moving towards the intersection node (i.e. in the opposite direction of the arrows). Right: selection of traces (see text for details).

We are interested in vehicles moving towards the intersection node, then only GPS traces *globally* moving towards the node are added up to the frame. More formally, the last record of the trace on the edge must be located closer from the intersection node than the first record (with respect to a distance metrics computed as a curvilinear abscissa along the edge geometry). Additionally, we required that the distance between both these extremal records is at least half of the edge length. For example, on the right part of figure 3, only traces 2 and 4 (solid lines) are taken into account in the frame generated from intersection n_1 (trace 1 is too short, while trace 3 is moving in the opposite direction). For the instance generated from node n_2, traces 1, 2 and 4 are discarded. Once traces moving towards a given node have been identified, we can extract sequences of GPS records corresponding to vehicle stops.

▶ **Definition 1** (Stop sequence)**.** Given a sequence of timestamped GPS points and two parameters: a maximal speed value $v_{max} \in \mathbb{R}^+$ and a minimal time duration $\tau_{min} \in \mathbb{R}^{+*}$, we define a *stop sequence* as a sub-sequence of consecutive records $S = \{(x_i, t_i) \mid p \leqslant i \leqslant q\}$ verifying the two following inequalities:

$$t_q - t_p \geqslant \tau_{min} \qquad \text{and} \qquad \forall\ i \in [\![p, q-1]\!] \quad \frac{|x_{i+1} - x_i|}{t_{i+1} - t_i} \leqslant v_{max}$$

where x is the curvilinear abscissa of GPS records along the edge. Simply put, for being qualified as a stop sequence, a portion of trajectory must be slow enough for a sufficiently long period of time. Also, note that p and q must be chosen in such a way that it is impossible to add new records to the sequence without breaking the inequalities stated above.

▶ **Definition 2** (Stop point)**.** For a given stop sequence, a *stop point* is defined as the mean position of points in the sequence, associated with the total duration of stop.

For each instance, stop points have been extracted from the selected traces according to definitions 1 and 2 with the following parameters: $v_{max} = 0.5\ m.s^{-1}$ and $\tau_{min} = 5$ seconds.

Since the number of stop points is unpredictable, it is not a reasonable solution to train a classifier with a predefined number of stop points. Indeed, this solution would fatally imply that no prediction can be made on instances with too few stop points (for example in remote parts of the road map). Reversely, if too many stop points occurred on a given instance, there is no alternative but randomly selecting the appropriate number of stops to make it fit the model of classifier. A practical solution to this issue, is to estimate the distribution of stop durations along the road curvilinear abscissa with an adapted version of the kernel distribution estimation (KDE) method [22].

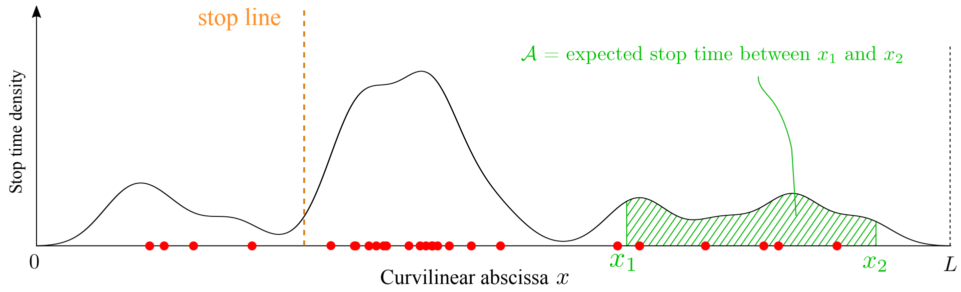

Figure 4 Weighted kernel density estimation of stop points. Orange vertical dashed line stands for the position of the stop line associated to a traffic signal (controlling the entrance on an intersection located on the left of the graphics). Vehicles are moving from the right to the left.

Let K be a positive, real-valued symmetric function whose integral sums up to 1. Function K is called a kernel. Let $x_i \in [0, L]$ be a set of N stop point locations, associated to stop duration times $t_i \in \mathbb{N}$ (for reasons that will be detailed further, we assume that timestamps are precise up to the second, which means that stop durations may be considered as integers). We define the *weighted kernel density estimation* as :

$$\forall\ x \in [0, L] : \qquad \hat{f}_h(x) = \frac{1}{Nh}\sum_{i=1}^{N} t_i K\Big(\frac{x - x_i}{h}\Big)$$

Note that this definition is slightly different from the standard KDE method, insofar as each kernel function centered in x_i is weighted by the corresponding stop duration t_i. As a consequence, $\hat{f}$ is not normalized:

$$\int_0^L \hat{f}_h(x)dx \simeq \int_{-\infty}^{+\infty} \hat{f}_h(x)dx = \frac{1}{N}\sum_{i=1}^{N} t_i \int_{-\infty}^{+\infty} K(x - x_i)dx = \frac{1}{N}\sum_{i=1}^{N} t_i = \mathbb{E}[t]$$

where $\mathbb{E}[t]$ is the expected stop time of all vehicles in the frame (this holds provided that the bandwidth parameter h is small in front of the instance dimension L). Similarly, as illusted on figure 4, the integral of $\hat{f}_h$ over a given segment $[x_1, x_2]$ is equal to a theoretical amount of time vehicles are expected to stop between curvilinear abscissa x_1 and x_2. Four examples of stop time distributions are depicted on figure 5 below.

Following a methodology inspired by [9], the resulting function is sampled at n evenly spaced locations to form the explanatory variable vector $X \in \mathbb{R}^n$. Eventually, target variables are computed. Binary classification variable $Y_1 \in \{0, 1\}$ denotes the presence of a traffic signal in the instance. If $Y_1 = 1$, regression variable $Y_2 \in [0, L]$ specifies the stop line location, measured as its distance to the intersection node (stop line abscissa on figure 4).

From a practical viewpoint, since we assumed stop durations are integer values, $\hat{f}_h$ may be computed with any standard KDE library, simply by oversampling data in such a way that each point x_i is present a number t_i of times. Besides, given that in efficient implementations of KDE, computation is done with Fast Fourier Transform algorithm, it makes sense to set the numerical value of n as a power of 2. In our application case, we took $n = 64$. Though it may be demonstrated that the mean integrated squared error is minimal with Epanechnikov kernel, the choice of the kernel function is not critical. Therefore we used a gaussian kernel. The bandwidth parameter has been set independently for each instance, according to Silverman's rule [22], which is optimal for normally distributed observations.

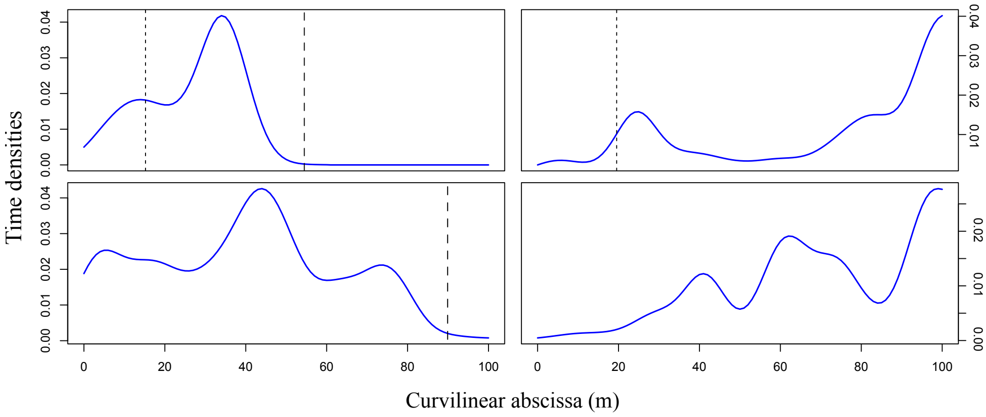

Figure 5 Examples of stop time distributions: the top two instances are positive (dashed line indicates traffic signal position), while the bottom two are negative. When the edge is shorter than 100 m, the thick dashed line denotes the end of the edge segment.

3.2 Training and validation

Given a set $\mathcal{D}$ of training instances in $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} = \{0, 1\}$ denote input and output spaces, respectively, and a new feature vector $X_u \in \mathcal{X}$, whose label Y_u is unknown, the task of a classifier is to estimate the probability of a traffic signal presence $\mathbb{P}(Y_u = 1|X_u, \mathcal{D})$. X_u is classified as positive whenever the estimated value is greater than 0.5. For regression problems, $\mathcal{Y} = \mathbb{R}$, and the objective is to estimate the conditional expectation $\mathbb{E}[Y_u|X_u, \mathcal{D}]$.

Introduced by Breiman [6] two decades ago, Random Forests (RF) algorithm is a statistically robust version of decision trees, relying on ensemble method concept to reduce prediction variance of individual decision trees. Given a collection of T decision trees whose posterior probability estimate is P_t, the overall posterior estimation is calculated as an average of predictions made by each individual tree:

$$\mathbb{P}(Y|X) = \frac{1}{T}\sum_{t=1}^{T} P_t(Y|X)$$

This makes Random Forests a simple, fast and efficient classification and regression tool, often considered as robust to over-fitting and particularly useful in high-dimensional problems where one has no strong reason to believe that all features will be helpful for discriminating instances. Moreover, in his foundation paper, Breiman introduced as well parameters setting empirical rules, which makes the tuning process quite straight-forward. For more detailed information about RF, we recommend the complete and extensive works of Louppe [15] for the theoretical background or Criminisi *et al.* [8] for a presentation of some of its capabilities in a wide range of practical problems.

The final dataset contains 4611 instances, including 662 (14%) positive samples. While the entire dataset is not overwhelmingly labeled as negative, this significant imbalance in favor of negative instances may markedly penalize the training process [2]. To overcome this issue, we tried different strategies: down-sampling (randomly suppressing negative samples until dataset is balanced) and up-sampling (replicating positive samples: this second strategy has the advantage of keeping all the information available from the data, at the expense of increasing correlation between individual samples). We also tried SMOTE algorithm [5], which is similar to up-sampling, but instead of replicating the minority class examples, new examples are generated by interpolation between randomly sampled neighbor instances of this class. We used $T = 500$ trees, and at each split $\sqrt{n} = 8$ features are taken into account. The model was validated by 10-fold cross validation, i.e. by training the algorithm on 90 % of the data, and validating it with the remaining 10 %, and repeating this process 10 times.

3.3 Inference on crossings

Given an intersection between a number n of incoming streets, each of them being classified by RF with a probability p_i of containing a traffic signal. We know that since the aggregated prediction relies on non-independent trees, and aggregation is calculated with a sum instead of a product, the values p_i are not strictly speaking probabilities. However, using the belief theory and Dempster-Shafer combination rule, it can be demonstrated through recurrence on n that the total belief towards the presence of a traffic signal on the intersection is:

$$\pi(p_1, p_2...p_n) = \prod_{i=1}^{n} p_i \times \Big(\prod_{i=1}^{n} p_i + \prod_{i=1}^{n} (1 - p_i) \Big)^{-1}$$

The intersection is then classified as controlled by a traffic signal when $\pi \geqslant \frac{1}{2}$. Using this combination rule, we may aggregate predictions on individual streets into a unique probability on the entire crossing, trading granularity for precision.

4 Results and discussion

The whole experimental process has been implemented in R with *randomForest* package [13] and launched on an Intel Core(TM) i7-3770 processor (3.40 GHz RAM 8 Go). We computed the following performance scores: specificity (or 1 - false positive rate, which corresponds to the recall), sensitivity (or true positive rate), area under receiver operating curve (AUC), training time (for a single fold, i.e. on 90% of the data), and overall accuracy.

Table 1 Detection performance scores for different way of balancing data.

Scores	Down-sampling	Up-sampling	Imbalanced	SMOTE
Specificity (%)	87.10	95.97	97.23	95.87
Sensitivity (%)	83.25	63.34	57.18	63.98
Accuracy (%)	86.57	91.49	91.73	91.49
AUC (%)	91.38	91.52	91.26	91.25
Training time (s)	1.35	6.98	3.83	7.18
Number of instances	1191	7108	4149	7108
OOB error rate (%)	14.46	2.00	8.23	2.36

Note that RF algorithm provides a practically unbiased error estimate during training phase (without validation dataset), called out-of-bag (OOB) estimate. Indeed, since training data are bootstrapped before used to grow decision trees, for a sufficiently large number of training data, it can be demonstrated that on average, each sample is not seen by a fraction $(1 - 1/n)^n \sim e^{-1}$ of trees. As a direct implication, each instance may be used as a training data for 63 % of trees, and passed through validation with the 37 % remaining trees.

From table 1, we observe that, as expected, the time complexity of the training process is roughly proportional to the number of training samples. Besides, area under curve (and then the overall performance) does not seem to depend upon the method selected for balancing the

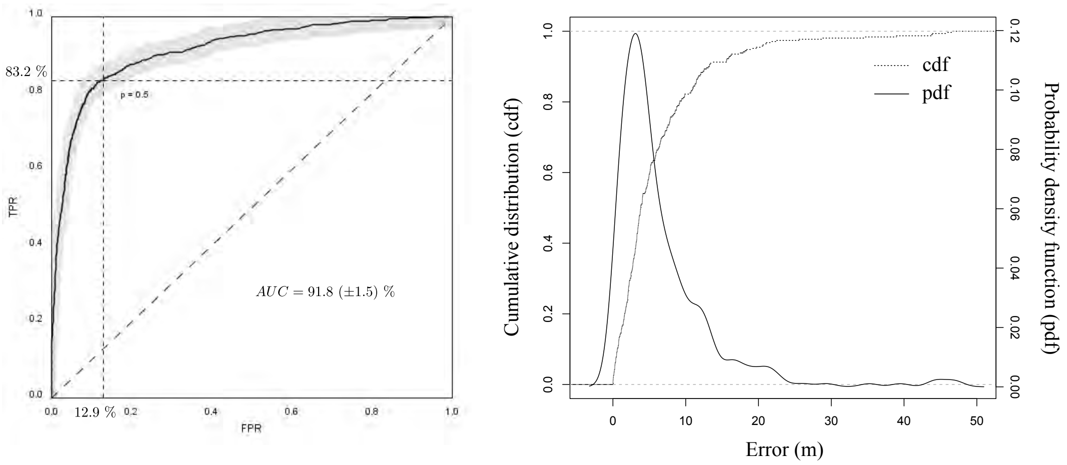

Figure 6 Left: Receiver Operating Characteristics (ROC) curve of the classifier with 95% confidence bands (computed with bootstrap method). Right: probability density and cumulative distribution functions of regression errors.

data. Everything happens just as if the four classifiers above correspond to different selection threshold of the same classifier model. Therefore, in the remaining of this section, we will only use down-sampling since it decreases the number of instances to process, resulting in a minimal computation time. It is worth noticing, that while OOB estimate is often acknowledged as being quite reliable, it completely fails to provide realistic error estimate on up-sampling and SMOTE experiments. This may be explained by the fact that with these two balance procedures, two identical (or at least very similar) sample data may be in and out-of-bag, which amounts to validating a model with samples partially contained in training dataset.

Figure 6 depicts detection and localization performances for the down-sampling version of the algorithm. Area Under Curve index of the classifier is equal to 91.8 (±1.5) % which is considered as a fairly good result. Though specificity is not so high (compared to the number of potential false positive that might be detected on a typical road network), the ROC curve is remaining close to the *no false positive* vertical line even for decent value of true positive rate. This observation instills confidence in the possibility of building a semi-automatic process, achieving a satisfying recall, and entailing only few manual corrections. However, on the other side of the ROC curve, it seems difficult to get all traffic signals, without spending a lot of time separating true and false positive detections. From a more practical viewpoint, it is also worth noticing that our recall may be compared with OSM (with the substantial advantage that our algorithm performs detection on each individual traffic signal, not only on the entire crossing).

Table 2 Localization performance scores. RMSE: root mean square error.

Scores	Mean error	Median error	Mode of errors	RMSE
Estimate (m)	6.22	3.82	2.65	9.51
Std. deviation (m)	±0.4	±0.3	±0.3	±0.8

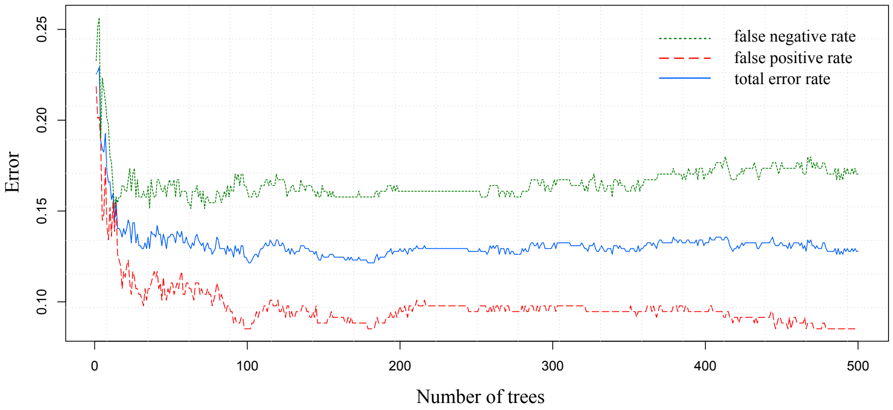

Figure 7 Out-Of-Bag (OOB) error estimate convergence versus number of trees T.

Besides, as depicted on the cumulative distribution function of regression errors, 82 % of errors are below 10 m, 60 % below 5 m, and 14 % as precise as 1 m. The root mean square error equals 9.51 (±0.8) m, (which is to be put in perspective to the 20 m of the standard deviation of the explained variable before regression), while mean, median and mode values are much lower, indicating that the distribution is significantly right-skewed. This calls for a more general discussion over what *detection* means. It might be more reasonable to count outliers as undetected (a stop line detected with 50 m inaccuracy cannot be legitimately considered as detected), as a result, the recall would decrease slightly by 4 % and as a reward, the RMSE of localization drops to 6 m, and mean error to 4 m.

Similarly to many ensemble method algorithms, RF is robust to overfitting, and while there is no guidelines for selecting an adequate number of trees, it is admitted that an excessive number is not harmful to the prediction (at the expense of an additional burden in computation time at training and inference steps). Figure 7 depicts the evolution of the OOB error estimate as trees are grown in the model. It may be observed that the convergence of predictions has been reached with approximately 100 trees.

Detailed inspection of the results revealed that many false detections occurred on places where very few vehicles traveled, which implies that the algorithm has not reached convergence as far as the number of vehicles is concerned. With a more extensive dataset we could certainly get better results. It would be interesting, in future works, to study the impact of the number of traces on the prediction scores.

A limitation of our work is that, as stated in section 3.1, our choice of frame, located near the intersection node, makes it impossible to detect traffic signals located in the middle of edges. Indeed, a relatively important number of errors occurred on traffic signals activated by pedestrians push button. An interesting proposition to solve this issue would be to up-sample the network by creating artificially dummy nodes evenly spaced on long edges. This approach may be successful to capture the remaining traffic signals. Another strong limitation of this work is that only information extracted from GPS traces upstream of the intersection is used to create the features, although the behavior of drivers downstream of a traffic signal may exhibit some very specific pattern that could help discriminate from stop signs at jammed intersections.

Based on the posterior probability values estimated by the RF, and combining them with the method proposed in section 3.3, we classified crossings into two categories, depending on whether they are controlled by a system of traffic lights. This made sensitivity and specificity increase to 87.9 % and 96.2 %, respectively, which is more than 8 % improvement in comparison to the per individual traffic light detection. This compares advantageously to OSM traffic signal database, particularly in terms of recall. Yet, specificity is not high enough to ensure fully automatic process without human supervision or post-processing corrections. Future research will try to leverage this correlation to improve results, even at the level of individual traffic signals. This can be done through relational learning techniques [21] and probabilistic graphical models [11], especially since we have a natural definition of network: the road map.

Apart from tuning more thoroughly the model parameters and the choice of features (additional data would preclude from over-fitting), among the main perspectives of improvement, we may attempt to use functional data analysis to decompose time distribution on an *ad hoc* basis of functions (e.g., wavelets, Karhunen-Loève transform...), in an attempt to minimize correlation between features. Extracting some other physical parameters such as speeds, accelerations, jerks... may also help discriminating traffic signals, as well as localizing it more precisely. This is possible, provided that GPS data speed profiles are smooth enough. Eventually, we may consider building *spatio-temporal* feature vectors, with a bi-dimensional kernel density estimation, where one dimension is the stop time and the second dimension is the stop position along the road axis.

5 Conclusion

Floating Car Data have been used so far in a wide variety of applications to infer the road network and its attributes. However, to the best of our knowledge, the method proposed in this paper is the first attempt to use multiple probe vehicle GPS traces along with statistical learning techniques to detect and localize traffic signals. Learning on a weighted-time distribution of stop points can reach up to 85 % detection scores, and approximately 5 m in positional accuracy. These results are promising for the future development but it is not yet sufficient at the moment to be used as a fully automatic detection system. Nonetheless, this algorithm might find some applications as it is, as a semi-automatic map inference algorithm with human post-process corrections, or when combined with other sources of data (e.g., sensors, embedded cameras, aerial images...) to provide a refined estimation with multi-source data fusion techniques. Future works will aim at improving detection scores by extracting more features from the data, and at extending this approach to other kinds of infrastructure elements. In the long run, one of the main prospects for this research is unquestionably autonomous cars, which, in addition to self-driving, would be self-mapping their environment and sharing information in a completely autonomous loop.

References

1 Cindie Andrieu, Guillaume Saint Pierre, and Xavier Bressaud. Estimation of space-speed profiles: A functional approach using smoothing splines. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 982–987. IEEE, 2013.

2 Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29, 2004.

3 James Biagioni and Jakob Eriksson. Inferring road maps from global positioning system traces: Survey and comparative evaluation. *Transportation Research Record: Journal of the Transportation Research Board*, 2291:61–71, 2012. doi:10.3141/2291-08.

4 Olivier Bonin. *Modèle d'erreurs dans une base de données géographiques et grandes déviations pour des sommes pondérées ; application à l'estimation d'erreurs sur un temps de parcours.* Thèse de doctorat, spécialité mathématiques - statistique, Université Paris VI - Pierre et Marie Curie, mar 2002.

5 Kevin W. Bowyer, Nitesh V. Chawla, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *CoRR*, abs/1106.1813, 2011.

6 Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

7 Yihua Chen and John Krumm. Probabilistic modeling of traffic lanes from gps traces. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pages 81–88, November 2010. `doi:10.1145/1869790.1869805`.

8 A Criminisi, J Shotton, and E Konukoglu. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. *Microsoft Research Cambridge, Tech. Rep. MSRTR-2011-114*, 5(6):12, 2011.

9 Mohamed el Habib Boukhobza and Malika Mimi. Classification automatique de la densité des tissus mammaires. *Traitement du Signal*, 33:441–460, 2016.

10 Alireza Fathi and John Krumm. Detecting road intersections from gps traces. In *International Conference on Geographic Information Science*, pages 56–69. Springer, 2010.

11 Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

12 Jun Li, Qiming Qin, Jiawei Han, Lu-An Tang, and Kin Hou Lei. Mining trajectory data and geotagged data in social media for road map inference. *Transactions in GIS*, 19(1):1–18, 2015.

13 Andy Liaw, Matthew Wiener, et al. Classification and regression by randomforest. *R news*, 2(3):18–22, 2002.

14 Xuemei Liu, James Biagioni, Jakob Eriksson, Yin Wang, George Forman, and Yanmin Zhu. Mining large-scale, sparse gps traces for map inference: Comparison of approaches. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 669–677, New York, NY, USA, 2012. ACM.

15 Gilles Louppe. Understanding random forests: From theory to practice. *arXic*, 2014. `arXiv:1407.7502`.

16 Y Méneroux, D Manandhar, S Ranjit, G Saint Pierre, and R Shibasaki. Positional accuracy control in dense urban environment with low-cost receiver and multi-constellation gnss. In *Proc. 9th Multi-GNSS Asia – MGA Conference*, 2017.

17 Volodymyr Mnih and Geoffrey E. Hinton. Learning to detect roads in high-resolution aerial images. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 210–223, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

18 Ana Tsui Moreno and Alfredo García. Use of speed profile as surrogate measure: Effect of traffic calming devices on crosstown road safety performance. *Accident Analysis & Prevention*, 61:23–32, 2013.

19 Mario Munoz-Organero, Ramona Ruiz-Blaquez, and Luis Sánchez-Fernández. Automatic detection of traffic lights, street crossings and urban roundabouts combining outlier detection and deep learning classification techniques based on gps traces while driving. *Computers, Environment and Urban Systems*, 68:1–8, 2018.

20 Paul Newson and John Krumm. Hidden markov map matching through noise and sparseness. In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 336–343. ACM, 2009.

21 Prithviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Galligher, and Tina Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008.

22 Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

23 Mohit Dev Srivastava, Shubhendu Sachin Prerna, Sumedha Sharma, and Utkarsh Tyagi. Smart traffic control system using plc and scada. *International Journal of Innovative Research in Science, Engineering and Technology*, 1(2):169–172, 2012.

24 Leon Stenneth and Philip S. Yu. Monitoring and mining gps traces in transit space. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 359–368, 2013. `doi:10.1137/1.9781611972832.40`.

25 Karl Van Winden, Filip Biljecki, and Stefan Van der Spek. Automatic update of road attributes by mining gps tracks. *Transactions in GIS*, 2016.

26 Christopher K. H. Wilson, Seth Rogers, and Shawn Weisenburger. The potential of precision maps in intelligent vehicles. In *IEEE International Conference on Intelligent Vehicles*, pages 419–422. Citeseer, 1998.

27 Lijuan Zhang and Monika Sester. Incremental data acquisition from gps-traces. In *Geospatial Data and Geovisualization: Environment, Security, and Society; Special Joint Symposium of ISPRS Commission IV and AutoCarto*, 2010.

28 Qiaoping Zhang and Isabelle Couloigner. Automated road network extraction from high resolution multi-spectral imagery. In *Proceedings of ASPRS 2006 Annual Conference*, pages 01–05, 2006.