2017 IEEE 18th International Conference on Mobile Data Management

Probabilistic Speed Profiling for Multi-lane Road Networks

Bing Zhang
Northwestern University
2145 Sheridan Road
Evanston, IL, 60201
Email: bing@u.northwestern.edu

Goce Trajcevski
Northwestern University
2145 Sheridan Road
Evanston, IL, 60201
Email: goce@eecs.northwestern.edu

***Abstract*—We address the problem of incorporating uncertain location data in the generation of speed profiles for vehicles on roads with multiple lanes. Moving objects' location data can be obtained from different/multiple sources – e.g., GPS on-board the moving objects, roadside sensors, cameras. However, each source has inherent limitations that affect the precision – from pure measurement-errors, to sparsity of their distribution. Incorporating such imprecisions is paramount in any query/analytics oriented system that deals with location data. The difficulties multiply when one needs to reason about localization with lane-awareness and attempts to use the location-in-time data to enable effective navigation systems. To tackle this problem, we take a step towards: (a) incorporating uncertainty of the objects' locations into traditional map-matching processes, thereby augmenting them with its impact on different lanes; (b) introducing an information-theoretic distance function that can be used to decide when two "units" qualify to belong to a same cluster. Our experiments demonstrate that the proposed approach offers a more effective way to generate spatio-temporal clusters with similar speed profiles which, in turn, enables more efficient routes generation.**

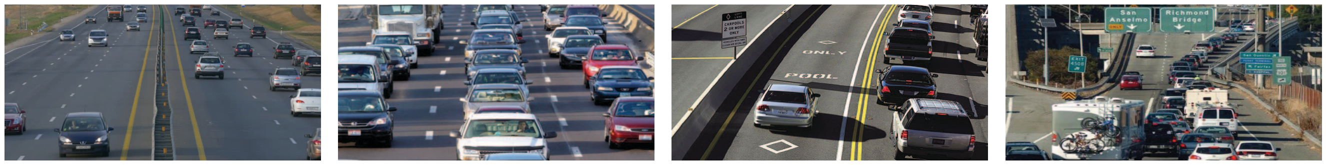
***Keywords*-Speed profiles, Uncertain trajectories, Multi-lane roads**

I. Introduction

Lane level positioning and navigation have been one of the challenging tasks that have spurred a significant amount of recent research since accurate navigation is at the very core of the autonomous driving [6], [29]. Models for lane-level high-definition maps have been proposed in different applications' settings [1], but lane-aware traffic inference and route planning are still investigated, mostly from two perspectives: (a) Assuming very accurate positioning data gathered through Differential GPS (DGPS) or laser scanners [28]. The high cost of sensors prohibits this method from being widely deployed for production cars; (b) Fusing heterogeneous data sources, i.e., combining GPS data with camera and using computer vision for lane recognition [6], [19]. The bottleneck of this approach is the speed of image processing, which constrains the use case in a real-time manner. Routing and navigation in modern traffic systems have been investigated since the 1980s [20], with techniques coming from both databases [14], [36] and transportation communities [26]. Typically, the algorithmic solutions rely on certain estimated values of the traffic flow – e.g., average speed – along the segments of the underlying road-networks, which vary dynamically [36] within a certain period (e.g., a day), depending on factors such as: time of day, capacity (lanes), road surface, etc.

From traditional vehicle routing problem [20] up to recent Eco-routing works [21], the methodologies (data properties, algorithms, etc.) proposed in various contexts share the assumption that on any road segment, at a certain time-period, vehicles have only one kind of a speed/motion. However, due to the multiple lanes, vehicles on the same road segment and at the same time instant/interval, may have different speeds. This, in turn, implies that using the average speed as a descriptor may not be good enough for many routing-based applications. Figure 1 illustrates four different real life scenarios on highways. When there are few cars on the road or the highway is fully congested, traffic speeds are relatively uniform among the lanes (cf. Figure 1a and 1b). However, Figure 1c, shows how a high-occupancy lane (also known as carpool lane; restricted traffic lane reserved during rush hour for the exclusive use of vehicles with one or more passengers) usually has higher speeds than the other lanes. Similarly, Figure 1d, shows a highway exit 450B on U.S. route 101 in California, near Richmond-San Rafael Bridge. The cars back up at the rightmost lane towards the bridge, while the left lane on northbound U.S. route 101 has very low densities. Thus, averaging the observations from particular (groups of) vehicles, could yield an inaccurate picture about the traffic distribution – and, yet, most of the popular traffic speed estimation methods are based on averaging the samples from vehicles over a period of time or area — e.g., *Time Mean Speed* and *Space Mean Speed* [13].

At the heart of the motivation for this work is the observation that – to the best of our knowledge – the state of the art approaches have not provided solutions that would couple the multi-lane information with location uncertainties, when designing traffic speed profiles (we note that this is also the case for the existing works on map-matching GPS points from moving objects [5], [37]). Consider the following query:
Q1: *What is the distribution of the traffic speed on the route 101 between San Francisco and Richmond-San Rafael*



(a) Uniform traffic speed with low congestion (b) Uniform traffic speed with high congestion (c) Nonuniform traffic speed for carpool lane (d) Nonuniform traffic speed for highway exit

Figure 1: Traffic speeds vary among different lanes in different scenarios

Bridge, between 8:00AM and 10:00AM?

Traditional methods [25], [30], [36] would answer **Q1** with a single average value, possibly varying it throughout different time intervals between 8:00AM to 10:00AM (e.g., the average speed is updated every 30min.); and along different distances from Marine city on the route 101. However, this will yield incorrect values because the averaged speed will be applied within certain distances before/after exit 450B (cf. Figure 1d), yielding incorrect time-estimates for trip planning. As a complement to our previous work [39], we propose an approach for multi-lane speed pattern mining framework which addresses the aforementioned challenges and incorporates the location uncertainties due to GPS errors. The main contribution of this work can be summarized as follows:

- We propose a novel probabilistic model to represent location uncertainties and apply it to spatial temporal data mining.
- We propose a novel distance function and an improved speed cluster mining algorithm for multi-lane road networks.
- We present experimental observations conducted on the Rome Ring Road to demonstrate the benefits of the proposed approaches.

II. Preliminaries

We now present a brief overview of the related background and introduce the basic terminology.

Traditionally, in MOD [16] the motion of an object with a distinct ID (*oID*) is represented as a *trajectory* $Tr_{oID} = [p_1, p_2 ... p_n]$, where each point p_i is a triplet $p_i = (x_i, y_i, t_i)$; t_i being the time that the object was at location (x_i, y_i).

A *road segment* r is a octuple $r = (r_{ID}, r_{Dir}, r_s, r_e, r_{type}, r_{length}, r_{speed}, r_{lane})$, where: r_{ID} is its unique identifier; r_{Dir} is a binary value indicating whether r is one-way or two-way segment; r_s and r_e are k-tuples (k = number of lanes) representing the starting and ending points of each lane (centroids); r_{type} indicates the type of the road to which the segment belongs (e.g., urban, rural, etc...); r_{length} is its length; r_{speed} is the maximum speed; and r_{lane} is an integer specifying the number of lanes in each direction. A *road network* is an (augmented) graph $G_{RN} = (V_{RN}, E_{RN})$ where V_{RN} is the set of nodes representing the terminal points of road segments, and E_{RN} is the collection of road segments.

Table I: Lane Width for Different Types of Road

Type of Roadway:	Rural	Urban
Freeway	12ft	12ft
Ramps (1-lane)	12-30ft	12-30ft
Arterial	11-12ft	10-12ft
Local	9-12ft	9-12 ft

For types of roads and the width of the lanes, we assume the classification proposed by the FHWA (Federal Highway Administration) of the US Department of Transportation [11] illustrated in Table I, noting that the width is often associated with the maximum prescribed speed limit. Traffic-stream studies use different measures to characterize motion along road segments [13], often coupled with the available technology. For example, inductive sensors are good for estimating the flow, however, they cannot characterize the speed. On-board GPS devices are good at obtaining an average speed of individual moving objects, however, they are error-prone in terms of location, and cannot capture fluctuations in-between samples. We assume that motion-relevant data is obtained from (a sequence of) GPS points.

In information theory, a classical measure of information in a stochastic setting is the *Shannon Information* [17]. The Shannon Information $S_P(A)$ (also called the surprisal, or self-information) of a probability distribution P for an event A is $S_P(A) = -log_2 P(A)$. The *information entropy* (also called Shannon Entropy) is the expected value of the Shannon information [31].

A. Path Based Map-matching

Map-matching algorithms use information generated from positioning technologies and supplement with data from a high resolution spatial road-network map to provide an enhanced positioning output. It identifies the the correct road on which vehicles travel and determines vehicles' location on that segment [27]. Map-matching approaches can be generally categorised into four groups: geometric, topological, probabilistic, and other advanced techniques [27].

In this paper, we apply and implement a path based map-matching algorithm that uses a Hidden Markov Model (HMM) to find the most likely road route [23]. Compared with the traditional point-based map-matching algorithm that only utilizes the geometric information from GPS points, the path based map-matching take the connectivity relationship between consecutive GPS points into consideration. It also uses Viterbi algorithm to compute the global optimal path.

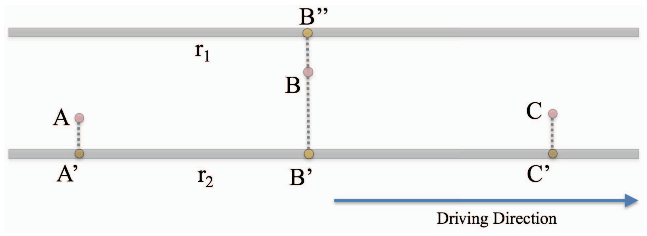


Figure 2: Path based map-matching

Figure 2 is a zoom-in view of the highway road network, which is the environment we developed for experiment. There are two parallel, directed edges r_1 and r_2 representing two directed road segments of the highway. One car is driving counterclockwise on r_2, forming a trajectory $[A, B, C]$. The correct map-matching results are $[A', B', C']$. When applying point-based map-matching algorithm, that matches GPS points to line segments with smallest matching distance [41], point B is map-matched to B'' due to the GPS noise and the factor that point B is closer to road segment r_1. However, trajectory $[A', B'', C']$ is invalid since a jump between r_1 and r_2 is not allowed. Path based map-matching incorporate geometric and topological information between every two consecutive GPS points, and is able to correct the map-matching result for point B from B'' to B'.

B. *Partitioning*

The philosophy of clustering traffic speed data is to group those GPS points that are spatially and temporally close to each other *and* with similar speed. The Unit Cell (UC) is defined as [39]:

Definition 1. (*Unit Cell*): A Unit Cell $UC_{kl} = (\Delta^S_{kl}, \Delta^T_{kl}, V_{kl}, D_{kl})$ in the l^{th} lane of a given road segment is the minimal partition in spatial and temporal dimension, characterized by a spatial range $\Delta^S_{kl} = d^+_{kl} - d^-_{kl}$, temporal interval $\Delta^T_{kl} = t^+_{kl} - t^-_{kl}$, and a set of trajectories $D_{kl} = [Tr_1, Tr_2, ...Tr_n]$ that belong to it. The set D_{kl} determines the speed-value V_{kl} associated with UC_{kl}

We note that the spatial range uses only “1D interval” – i.e., $d^+_{kl} - d^-_{kl}$ because the “conventional” 2-D space is constrained to 1-D along the driving direction, representing the distance(s) from starting point of the road segment (for the corresponding lane) until the beginning of the k-th unit cell (and the width is pre-determined by the road-type).

It is possible that multiple UCs share the similar speed/speed cluster(if we regard all GPS points within one UC as a single cluster). A Merging Cell (MC) is the combination of two or more UCs.

Definition 2. (*Merging Cell*): A Merging Cell (MC) is a union of multiple neighboring unit cells $MC_j = UC_1 \cup UC_2 \cup ... \cup UC_n$. Its spatial range is defined as $R_{MC} = \cup_i \Delta^S_i$ and its temporal interval $T_{MC} = \cup_i \Delta^T_i$.

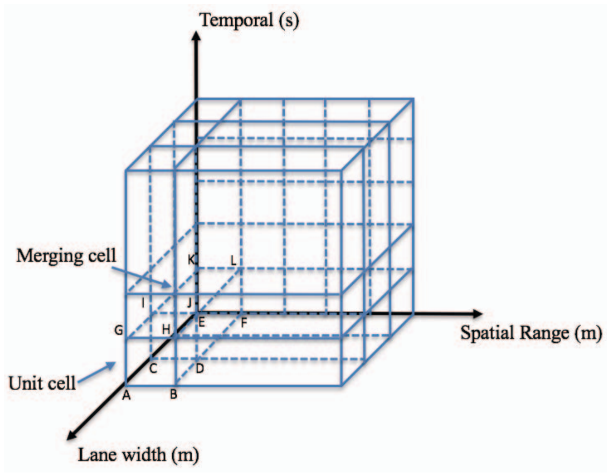


Figure 3: Unit cell and Merging cell

The merging process follows certain criteria and procedures, like speed threshold and agglomerative merging in the previous work [39]. In the following sections, we will propose a new distance measurement and merging algorithm.

III. Probabilistic GPS Model and Speed Profile

GPS devices yield measurement error associated with each GPS-based determined location, even more so for crowd-sourced GPS data collected from potable devices. Due to the constraint of device size and cost, average horizontal errors from consumer-grade GPS receiver range from few meters to tens of meters [38]. Thus, deterministic lane level computations based on GPS probe data are ambiguous, so much so that the position may yield a different lane. In this section, we first introduce a probabilistic model to describe the location whereabouts for GPS points, followed by a definition of speed profile for every UC.

A. *Probabilistic GPS Weight*

The uncertain disk model [33] is the most naive one for uncertain location data, assuming uniform distribution. Let $D_p(x, y, t, r)$ denote the disk centered at point $P(x, y, t)$ with radius r, and A_i denote the area of lane i, which is a rectangle shape area. The probability of a GPS point located within a certain lane can be estimated by: $P_{lane} = \frac{D_p(x,y,t,r) \cap A}{D_p(x,y,t,r)}$. More sophisticated models describe the GPS data as a zero-mean Gaussian model [23], [34]. In this paper, we do not consider the GPS errors along vertical axis, thus, measurements from GPS receivers follow 2D Gaussian model. Given a GPS point $P_i(x_i, y_i, t_i)$, the probability density function (pdf) is:

$$f(x, y) = \frac{1}{\pi \sigma_x \sigma_y} exp(-(\frac{(x - x_i)^2}{2\sigma_x^2} + \frac{(y - y_i)^2}{2\sigma_y^2})) \quad (1)$$

The 2D Gaussian model is illustrated in Figure 4, with lane-width of 5m and spatial range Δ^S_{kl} is 5m as well.

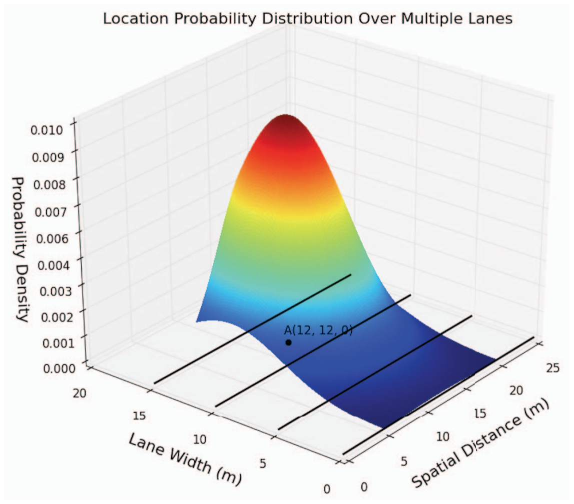


Figure 4: GPS point location probability distribution

The location pdf for GPS point $(12, 12, 0)$ spans over multiple UCs and the probability of the GPS point being located into one UC is the integral of density function 1 over the spatial range within the lane width. Given $UC_0(\Delta^S_{k0}, \Delta^T_{k0}, V_{k0}, D_{k0})$ with spatial range $[x_{uc}, x_{uc} + \Delta^S_{k0}]$ and lane width $[y_{uc}, y_{uc} + width]$, the probability of P_i being inside UC_0 is:

$$P_i = \int_{x_{uc}}^{x_{uc}+\Delta^S_{k0}} \int_{y_{uc}}^{y_{uc}+width} f(x, y)dydx \tag{2}$$

Definition 3. GPS Contribution C_{GPS}: A GPS point G contributes to a Unit Cell UC when the probability P for G being located in UC is greater than ω – a threshold for the minimum probability value.

We augment each trajectory $Tr_{o_{ID}} = [p_1, p_2...p_n]$ within a UC by a GPS contribution, thereby making each p_i a quadruplet $p_i = (x_i, y_i, t_i, C_{GPSi})$.

B. Speed Profile

Given a UC with a set of augmented supporting trajectories, we apply a discretized histogram to estimate the traffic distribution within each UC. Given a bin size ϕ and number of bins n, instead of counting, the frequency for each bucket is the aggregation of contribution from every GPS point, reflecting a weighted sum for different points. The number of bins determines the level of granularity of the speed profile. We note that using only one bin is equivalent to a single average speed while too many bins may incur computational overhead. For a set of GPS points S with speed ranges within $(i * \phi,\ (i + 1) * \phi)$, the total GPS contribution W is:

$$W(i) = \sum_{s} C_i \tag{3}$$

The discrete pdf for the speed within range $(i * \phi,\ (i + 1) * \phi)$, which we call a *Speed Profile*, is:

$$pdf_v(v \in (i * \phi, (i + 1) * \phi)) = Pr(i) = \frac{W(i)}{\sum_1^n W(i)} \tag{4}$$

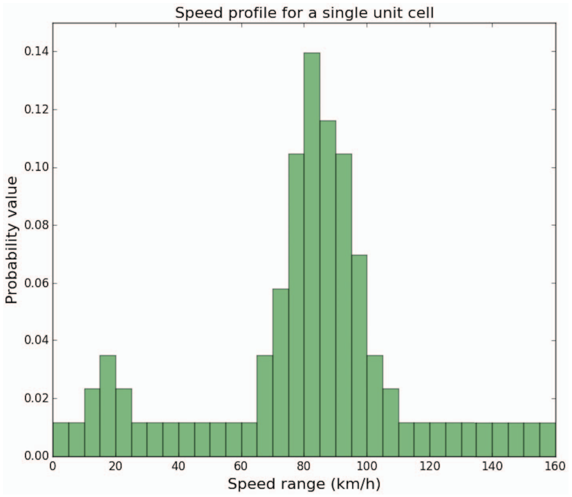


Figure 5: Speed profile for a unit cell

An example of a speed profile is shown in Figure 5, where ϕ is set to be 5km/h. There is one large peak at 80km/h, representing majority of GPS contributions; and a small peak near 20km/h, due to the contributions from slow lanes.

C. Merging Multiple Speed Profiles

Merging multiple UCs is the process to consolidate data from multiple sources, and we assume that the sources for GPS data within each UC are independent, because each UC is unique in spatial-temporal space and GPS measurements are independent. There are many approaches to consolidate independent data sources [18], like *averaging the probabilities*, and *averaging the data*, however, they have disadvantages: either do not take the differences of variances into consideration, or require averaging of dissimilar data.

Conflation is a method for consolidating a finite number of probability distributions $P1, ..., Pn$ into a single probability distribution $Q = Q(P_1, ..., P_n)$ [17], denoted by $\&(P_1, ..., P_n)$. Given multiple UCs $(UC_1, UC_2...UC_m)$ with respective speed profile $(pdf_{v1}, pdf_{v2}...pdf_{vm})$, the merged probability distribution obtained via conflation is:

$$\begin{aligned} pdf_v^{MC}(v \in (i * \phi, (i + 1) * \phi)) &= Pr(i) \\ &= \frac{\prod_1^m pdf_v(i)}{\sum_{y \in n} \prod_1^m pdf_v(y)} \end{aligned} \tag{5}$$

It has none of the disadvantages of the two averaging methods described above and has many advantages and important properties [18]:

(1) Conflation is commutative and associative:

$$\&(pdf_1, pdf_2) = \&(pdf_2, pdf_1) \text{ and}$$
$$\&(\&(pdf_1, pdf_2), pdf_3) = \&(pdf_1, \&(pdf_2, pdf_3))$$

(2) Conflation is iterative:

$$\&(pdf_1, pdf_2, pdf_3) = \&(\&(pdf_1, pdf_2), pdf_3)$$

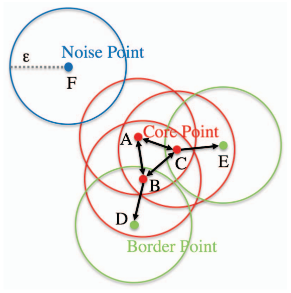
(3) Conflation minimizes the loss of Shannon information: If pdf_1 and pdf_2 are independent probability distributions, then the conflation $\&(pdf_1, pdf_2)$ of pdf_1 and pdf_2 is the unique probability distribution that minimizes, over all events, the maximum loss of Shannon information in replacing the pair pdf_1, pdf_2 by a merged distribution pdf^{MC} [18].

Properties (1) and (2) ensure that the conflation method can be used when merging UCs in any order and sequences, whereas (3) implies that conflation is compatible with entropy related measurement – i.e., given a threshold for the differences between two distributions in the information space, merged distribution using conflation minimizes the loss of Shannon information.

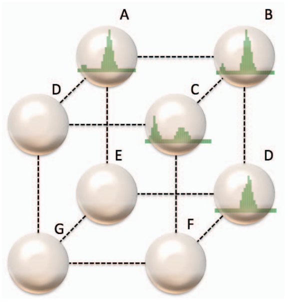
IV. Speed Cluster Mining

After finalizing the pre-processing of the uncertain GPS data and calculating speed profile for individual UCs, we now proceed with the mining steps – i.e., detecting the speed clusters in multi-lane settings.

We note that in our previous work [39] we proposed a sweep line based method for merging neighboring UCs by scanning along spatial, temporal and lane dimensions. While the algorithm proposed in [39] is effective in terms of reducing the number of UCs via merging, we observed that different merging sequences will generate different clustering results. To alleviate this phenomenon, in the rest of this section, we present an improved merging algorithm inspired by density based clustering.



(a) Traditional DBSCAN



(b) UC as "dummy point"

Figure 6: DBSCAN inspired merging

A. Probabilistic Distance Measurement

The most important criterion for merging two UCs is the level of similarity between them, in terms of the respective speed profiles. In the previous work [39], we used a simple method which compared the average speeds and the number of trajectories between two UCs with corresponding thresholds. However, in this work we use the speed profile as a more comprehensive description for drivers' behavior within certain UC and, given that we are catering to the fact of uncertainty of the locations' values, we decide whether to merge two UCs or not by calculating the distance between the respective discrete probability distributions. However, the distance between two distributions cannot be easily captured through geometric distance. Typically, in information theory, an uncertain object is treated as a random variable following a particular probability distribution. One popular measure for calculating the distance between two pdf's is the Kullback-Leibler divergence (also called information gain) – essentially, a measure of the difference between two probability distributions [22]. In our settings, given two speed profiles described with the respective pdf_i and pdf_j, the Kullback-Leibler divergence from pdf_i to pdf_j – denoted $D(pdf_i||pdf_j)$, describes the amount of information loss when pdf_i is used to estimate pdf_j. The equation defining the Kullback-Leibler divergence is:

$$D(pdf_i||pdf_j) = \sum_{v} pdf_i(v) log \frac{pdf_i(v)}{pdf_j(v)} \tag{6}$$

However, one specific property of the Kullback-Leibler divergence is its asymmetry, which is fine in many applications settings that rely on Bayesian inference. Contrary to this, in our settings, we would like to have the merging of UCs to be an undirected process that can start from any UC – and this makes the asymmetry an undesirable property.

Another method to measure the similarity between two probability distributions is the Jensen-Shannon divergence (JSD) (a.k.a. information radius) [22], which is commonly used in clustering probability distributions. It is based on the Kullback-Leibler divergence with the notable properties that it is symmetric, always a finite value and the square root is a metric. The Jensen-Shannon divergence between two speed profiles pdf_i and pdf_j is denoted as JSD_{ij}.

$$JSD_{ij} = \frac{1}{2} D(pdf_i|| \frac{pdf_i + pdf_j}{2}) + \frac{1}{2} D(pdf_j|| \frac{pdf_i + pdf_j}{2}) \tag{7}$$

In this work, we adopt the Jensen-Shannon divergence as the distance function between two UCs.

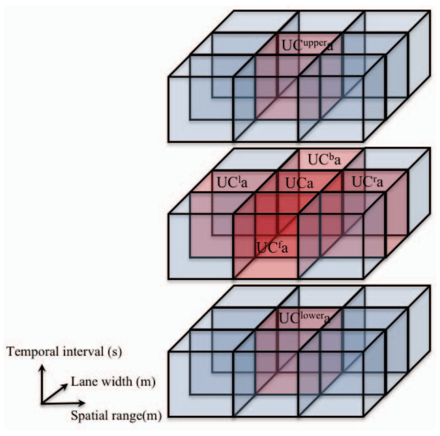


Figure 7: One UC has six neighboring candidates

B. Mining Speed Clusters

The traditional DBSCAN [10] accepts a radius value ε based on a (user defined) distance measure, and a value MinPts for the number of minimal points that should occur within Eps radius. A simple illustration is shown in Figure 6a, where $\varepsilon = 2$ and the Euclidean distance is used as a distance so that, upon comparison with ε, one can determine whether two points are connected. The points A, B and C in Figure 6a are considered core points because the discs with radii ε and centered at each of them, contain at least 2 neighboring points and they belong to the same cluster. Points D and E are not core points. However, they are reachable from A through other core points – and, consequently, they belong to the same cluster as well. The point F is a separated/isolated noise point that is neither a core point nor density reachable [10].

Inspired by DBSCAN, we propose Traffic-Density-Merging (TDM), a density based clustering algorithm. When mining the speed cluster based on fine-grained UCs, each UC is treated as an "artificial point", as shown in Figure 6b. They are well-identified in terms of their organization in the spatio-temporal 3D space and the Euclidean distances between two spatially-consecutive points are the same. The candidate-neighbors of a particular such "artificial point" are defined as UCs which have representative "artificial points" that are directly connected to the one representing the particular UC. Thus, each "artificial point" can have at most 6 candidate neighbors, as shown in Figure 7. As a specific example, in Figure 6b, the "point" B has 3 candidate neighbors. However, only "point" A and D have similar speed profiles compared with "point" B, whose information-distance *JSD* is smaller than the (assumed) threshold. Therefore, B has two neighbors and is a core point (assuming MinPts $= 2$). A, B and D can therefore be merged as a speed cluster. This process is analogous to the one occurring in the traditional DBSCAN – except, instead of the Euclidean distance, JSD is used to calculate the respective distances in the "information space" and a corresponding threshold λ is defined to determine whether two neighboring UCs belong to the same speed cluster.

Proceeding formally, and in the spirit of [10], the density based speed cluster is defined as follows:

Definition 4. (*Density-based speed cluster*): A cluster C is a non-empty subset of UCs satisfying the following "maximality" and "connectivity" requirements:

1) $\forall p, q$: if $q \in C$ and p is density-reachable from q with respect to (λ) and MinPts, then $p \in C$.
2) $\forall p, q \in C$: p is density-connected to q with respect to (λ) and MinPts.

The TDM starts with an arbitrary UC_a. If it has been visited, the iteration breaks and proceeds to the next UC. Otherwise, we call the *Neighbor-Query* to retrieve its qualified neighbors. As shown in Figure 7, UC_a has six candidate neighbors. Those candidate neighbors that have JSD less than λ with the currently considered UC (UC_a) become neighbors that are density reachable from UC_a. In the case that the number of such neighbors is larger than MinPts, UC_a is a core UC and a cluster is identified. If the number of neighbors is less than MinPts, it cannot form an independent cluster, and we keep this UC as a separate one.

Algorithm 1 Traffic-Density-Merging (UCSets, MinPts, λ)

```
1: ClusterID = NextID(NULL);
2: for UC IN UCSets do
3:     if UC.visited == True then
4:         continue;
5:     end if
6:     UC.visited = True;
7:     Neighbors = Neighbor-Query(UC, UCSets, $\lambda$);
8:     if Neighbors.size() $<$ MinPts then
9:         Point.CID = Separate;
10:    else
11:        Point.CID = ClusterID;
12:        Cluster = ExpandCluster(UCSets, Neighbors, UC, ClusterID, MinPts, $\lambda$);
13:        SpeedCluster.add(Cluster);
14:        ClusterID = NextID(ClusterID);
15:    end if
16: end for
17: Return SpeedCluster;
```

Algorithm 2 ExpandCluster (UCSets, Neighbors, UC, ClusterID, MinPts, λ)

```
1: Seeds = Neighbors;
2: Seeds.add(Point);
3: while Seeds.size() $>$ 0 do
4:     CurrentUC = Seeds.first();
5:     Seeds.pop();
6:     if CurrentUC.visited != True then
7:         CurrentUC.visited = True;
8:         Cluster.add(CurrentUC);
9:         Cluster.MergeSpeedProfile(CurrentUC);
10:        NewNeighbors = Neighbor-Query(CurrentUC, UCSets, $\lambda$);
11:        if NewNeighbors.size() $>$ MinPts then
12:            Seeds.Append(NewNeighbors);
13:        end if
14:    end if
15:    if (CurrentUC.CID == NULL) OR (CurrentUC.CID == Isolated) then
16:        CurrentUC.CID = ClusterID;
17:    end if
18: end while
19: Return cluster;
```

In Algorithm 1, *UCSets* is either the whole set of UCs

Algorithm 3 Neighbor-Query (UC, UCSets, AverageSpeed, λ)

```
1: CandidateNeighbors = UCSets.search();
2: for Neighbor in CandidateNeighbors do
3:     if JSD(Neighbor, UC) $< \lambda$ then
4:         Neighbors.add(Neighbor);
5:     end if
6: end for
7: Return Neighbors;
```

on certain road segment or a proper subset of them. MinPts and λ are parameters that are provided as input to TDM and they can be determined analytically or through experiments, based on a particular scenario.

When a new/unvisted UC is identified as a core cell, a new cluster is generated. Following that, the function *ExpandCluster* is invoked, for which the pseudo-code is presented in Algorithm 2, in order to expand the cluster based on the current UC and its neighbors. The cluster expansion process is essentially a depth-first kind of a search. A stack is used to store all the seed UCs. If the current UC is unvisited, we retrieve its neighbors using *Neighbor-Query* in the same way as described above. The qualified UCs are pushed onto the stack under the condition that the current UC is identified as a core cell. If the current UC has not been classified into any cluster or it is previously marked as *Separated*, we append it into the current cluster.

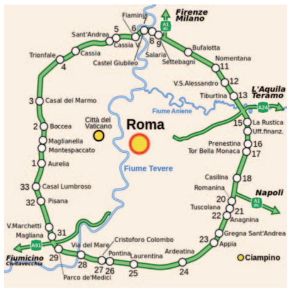
If two clusters C_1 and C_2 are very close to each other (in JSD sense), there might be scenario that a given UC_i belongs to both clusters – which entails that such UC_i is on the boundary between C_1 and C_2. If this is the case, UC_i will be assigned to the first discovered cluster. In addition, we note that there will not be cases in which a particular cluster partially intersects or if fully contained by another cluster (otherwise the two clusters will be merged).

Remark: The *Neighbor Query* can be supported efficiently by spatial access method like R*-trees [3], which is often available in spatial database system – however, the issue of indexing is beyond the scope of this paper and we defer it for our future work.

Given the results for the original DBSCAN, we note that the access time for a collection of n UCs is $O(logn)$. As we discover new clusters, for each of the n points there is at most one invocation of the Neighbor-Query to be processed. Therefore the time complexity for TDM is bounded by $O(nlogn)$ – which, once again, is the time complexity of the traditional DBSCAN (cf. [10]) since we retain the general framework for density based searching.

V. Experimental Observations

For our experiments, we used a data obtained from the Grande Raccordo Anulare (GRA) motorway. It is a toll-free, ring-shaped orbital motorway that encircles Rome, as illustrated in Figure 8a, and it is considered to be one of the most frequently used roads with heavy traffic for the most of the day. Our experiments are based on a dataset contains GPS traces of 320 taxi cabs in Rome, collected over 30 days – from February 1, until March 2 of 2014 [4]. The cardinality of the dataset is 8,368,858 points.



(a) The GRA

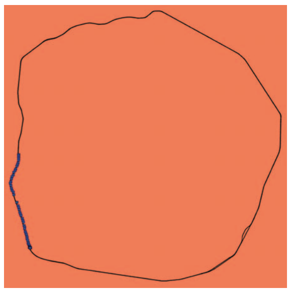

(b) Experiment environment

Figure 8: The Grande Raccordo Anulare and experiment environment

To evaluate the benefit of our proposed speed cluster mining algorithm, an augmented road network of GRA is built based on OpenStreetMap data, where road segments are augmented with three lanes. Figure 8b is a simple visualization of such augmented road networks. Blue dots represent GPS points that originated from two vehicles' trajectories driving along the GRA.

The experimental evaluation consists of two parts – training and validation; and there are two steps in the training process as well: (1) building speed profile for each UC, and (2) mining speed clusters. We first train the model from GPS traces collected from Rome taxi cabs. Subsequently, we validate our model by predicting the travel time using trained speed clusters.

The experiments were conducted on a MacOS machine with 2.7 GHz Intel CPU with 8GB 1867MHz DDR3 RAM and the implementation[1] was done in Python 2.7.

Parameter Estimation and System Implementation: Our probabilistic GPS uncertainty model calculating the GPS contribution for each UC requires two parameters σ_x and σ_y. They are the values of the corresponding standard deviation of the Gaussion GPS noise in longitudinal and latitudinal directions. This parameter can be affected by the measurement devices and measurement environment. According to experiments conducted/reported in the related literature [23], we estimate the standard deviation of Gaussion GPS noise to be 5 meters.

In the Traffic-Density-Merging algorithm, there are two important parameters – MinPts, which determines whether a given UC is a "core point", and λ – which is the threshold to determine the neighboring relationship between two UCs.

[1]We note that the code and the datasets are publicly available at www.eecs.northwestern.edu/˜bzv686.

MinPts is an integer in the range $[1, 6]$ and λ is a real number from the interval $[0, 1]$.

There are many different ways of choosing data structure to implement our proposed traffic speed cluster mining algorithm. In our experiments, we used a simple scheme – i.e., we built a three dimensional matrix in spatial-temporal coordinates to index UCs on each road segments, which is similar to the structure shown in Figure 3. The respective dimensions in spatio-temporal coordinates are indexed to UCs that are stored in a key-value map. While the overall efficiency is not a topic of this work, we note that the structure used in this experimental setup allows for a fast query processing when inferring the traffic speed from trained speed clusters.

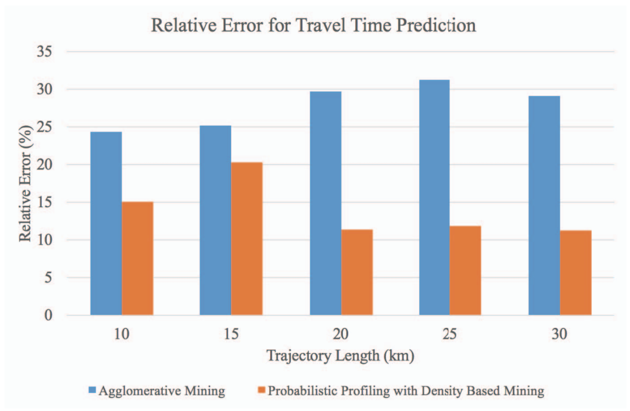


Figure 9: Relative error for predicted travel time

Estimated Travel Time Query: Our first set of experiments aims at illustrating how the speed clusters mining in the lane level granularity can provides a more accurate, yet compact description of the traffic distribution for road networks. Many applications could benefit from it – e.g., adaptive navigation, route planning and travel time prediction. In this experiment, we implemented the travel time prediction to demonstrate the advantage of our proposed method.

The Rome taxi dataset is divided into four folds according to sample time. Each of them contains GPS points within one week. We used three weeks data for training purpose, and the remaining one was used for validation.

GPS points in validation trajectories are assigned to corresponding lanes using the lane labeling process in [39]. Given a validation trajectory $Tr_{val} = [p_1, p_2 ... p_n]$, the traffic speed for certain GPS points $P_i = (x_i, y_i, t_i)$ can be inferred from the speed profile of the corresponding speed cluster. Thus, the predicted travel time $T_{predict} = \sum_1^{n-1} \frac{distance(P_i, P_{i+1})}{speed(x_i, y_i, t_i)}$. Since the ground true driving time $T_{true} = t_n - t_1$, we use relative prediction error $e = \frac{abs(T_{true} - T_{predict})}{T_{true}}$ to quantitatively measure the prediction power of the proposed model.

The baseline method we compared with is the agglomerative mining method we proposed previously [39]. It is a bottom-up clustering method with sweep line styled merging. The experimental results are shown in Figure 9. When the parameter MinPts is 3 and λ equals 0.1, the probabilistic speed profiling with density based merging method reduces prediction error by more than 20%. Compared with the simple agglomerative method, the new model describes the multi-lane traffic speed information with a probabilistic speed profile and is able to make a more accurate travel time prediction.

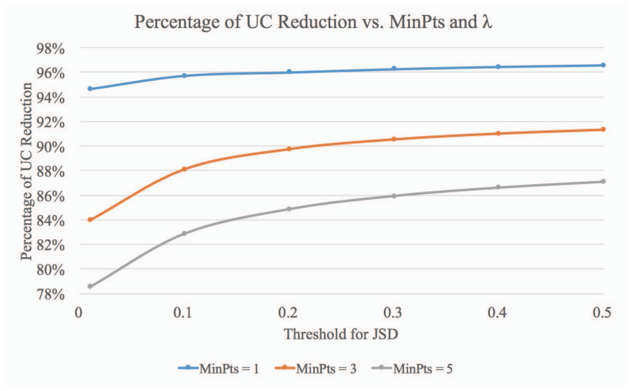


Figure 10: Percentage of UC reduction when changing parameter

UC Reduction: As described in Section 2.3, we partition the spatio-temporal space into fine-grained UCs. Since lots of them will share similar speed profiles, it is not necessary to store every UCs into the database. Therefore, merging is a beneficial operation in the cluster mining process. We reiterate that the sweep line styled merging, proposed in [39], has three possible merging directions for each UC and each direction is processed sequentially. The disadvantage for this merging method is its instability — the merging results are affected by different merging sequences. The density-based merging algorithm proposed in this paper overcomes this issue. Due to the nature of depth first search within the cluster expanding process, the density-based merging is independent of merging direction and merging sequences.

Figure 10 shows the percentage of UC reduction with various parameter choices. The highest compression ratio reaches more then 96%, while the lowest one is still more than 70% when MinPts is 5 and λ equals 0.1. The larger λ values and smaller MinPts values (which correspond to lessening the constraints for merging neighboring UCs) will incur higher compression ratio.

Training and Validation Time:In the last experiment that we report, we consider the respective execution times for the model training and validation. The system for this experiment is designed to run in an offline mode, where the speed clusters are mined from historical GPS data. As shown in Figure 11, the training process fo mining speed clusters takes relatively long time. We note that the training

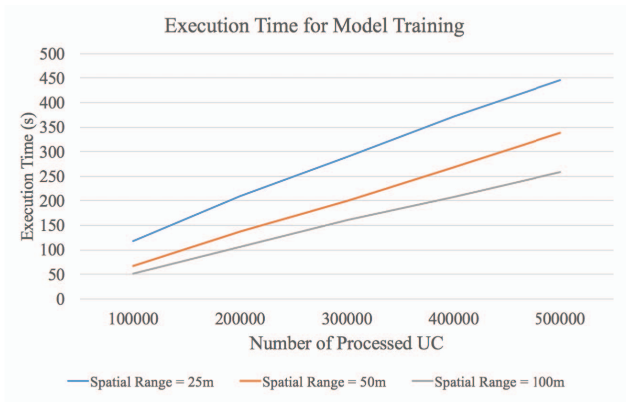


Figure 11: Execution time for speed cluster training

time is related not only to the size of the raw data, but also to the level of granularity of the partitioning. The smaller size of UC (i.e., more granular representation) yields a longer training time.

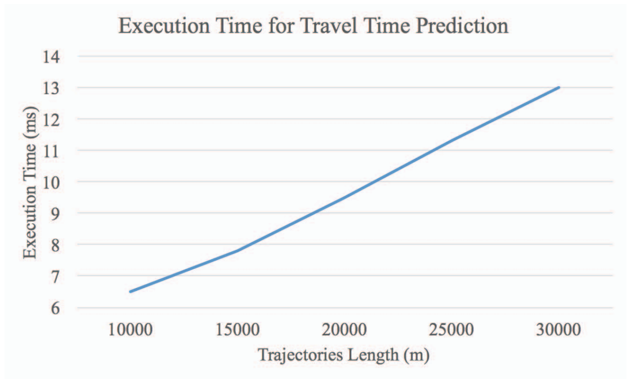


Figure 12: Execution time for travel time prediction

The main advantage for this offline trained model is that processing queries that depend on the traffic speed distribution is fast. When we validate the model by calculating predicted travel time, all queries are finished within a second. We note that this kind of an executional behavior is suitable for many OLAP kinds of applications.

VI. Related Work

Lane level positioning, routing and navigation are correlated research areas with high societal impacts. There are two main categories of related works in this realm. The first one attempts to directly map-match GPS points to corresponding lanes [8]. This method usually requires the use of Differential Global Positioning System (DGPS) for data collection so that the GPS errors are smaller than the usual lane width. However, because of the high cost, DGPS has not been widely available in consumer grade mobile devices. Other researchers choose to pursue external calibration through computer visions [35], vehicle-to-vehicle (V2V) communication or vehicle-to-infrastructure (V2I) communication [2], [9]. These approaches require additional hardware or infrastructure and cannot be applied in large scale quickly.

Multiple models have been proposed to answer queries related to GPS uncertainties. From disk model of location uncertainty (yielding sheared cylinder model in spatio-temporal space) [33], through beads model [32], to adaptation of the bead model on road networks [24]. More recently, an attempt to combine heterogeneous location data sources in the context of multi-lane road networks, called fused bead model was presented in [40].

A complementary body of related works stems from the literature addressing problems related to trajectories clustering, for both online and offline settings. Various clustering algorithms and frameworks have been proposed, including regression [12], partitioning and grouping [15] and density based clustering [7]. However, most of these works are targeting the, so called, macroscopic model and focus on large scale pattern mining, which lead to application like popular region discovery, event detection or route analysis. In addition, very few of them combine the trajectories clustering techniques with the constraint of road networks and use it as a tool to analyze the traffic on the lane level granularity.

VII. Concluding Remarks

We proposed a methodology for mining speed clusters in multi-lane road networks, incorporating the uncertainty of the moving objects location to capture the GPS errors within the model. We proposed a novel distance function and a variant of the DBSCAN algorithm for mining multi-lane speed clusters. We used the Rome taxi data to demonstrate that, compared with the agglomerative approach (cf. [39]), our proposed method yields both a more compact representation of the clusters, as well as a more accurate travel time calculation for trajectories.

There are several extensions to our work. Firstly, we plan to tackle several efficiency-related aspects – namely, data structures that will enable efficient storage and retrieval of the elementary UCs. Our next aim is to incorporate a few distinct contexts: (1) we would like to investigate the impact of changes in the type of the road (i.e., from 4 lanes expressway into a single lane local street); (2) we believe that the an attribute with a stronger impact may be the kind of a vehicle (e.g., passenger car vs. trucks); and (3) we plan investigate the impact of speed/travel-time clustering in the settings of multi-modal transportation. Our longer term vision is to develop a model that will balance the trade-offs between the precision of the clustering vs. the cost (both in terms of access as well as execution time), when multiple data sources can be combined – e.g., roadside sensors and cameras – with the GPS-based location data.

Acknowledgment: The work was supported by the NSF grants III 1213038 and CNS 1646107, and the ONR grant N00014-14-10215.

References

[1] Autonomous cars can only understand the real world through a map, 2016. https://goo.gl/Q7U1bw.

[2] N. Alam, A. T. Balaei, and A. G. Dempster. An instantaneous lane-level positioning using dsrc carrier frequency offset. *IEEE Trans. on Intelligent Transportation Systems*, 13(4):1566–1575, 2012.

[3] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. *The R*-tree: an efficient and robust access method for points and rectangles*. ACM SIGMOD, 1990.

[4] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, R. Amici, and A. Rabuffi. CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from http://crawdad.org/roma/taxi/20140717, July 2014.

[5] D. Chen, A. Driemel, L. J. Guibas, A. Nguyen, and C. Wenk. Approximate map matching with respect to the fréchet distance. In *ALENEX*, pages 75–83, 2011.

[6] D. Cui, J. Xue, and N. Zheng. Real-time global localization of robotic cars in lane level via lane marking detection and shape registration. *IEEE Trans. on Intelligent Transportation Systems*, 17(4):1039–1050, 2016.

[7] T. L. C. da Silva, K. Zeitouni, and J. A. de Macêdo. Online clustering of trajectory data stream. In *IEEE Mobile Data Management (MDM)*, pages 112–121. 2016.

[8] J. Du and M. J. Barth. Next-generation automated vehicle location systems: Positioning at the lane level. *IEEE Trans. on Intelligent Transportation Systems*, 9(1):48–57, 2008.

[9] J. Du, J. Masters, and M. Barth. Lane-level positioning for in-vehicle navigation and automated vehicle location (avl) systems. In *Proc. IEEE Intelligent Transportation Systems*, pages 35–40. 2004.

[10] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. ACM SIGKDD*, pages 226–231, 1996.

[11] AASHTO. *A Policy on Geometric Design of Highways and Streets*. 2011.

[12] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In *Proc. ACM SIGKDD*, pages 63–72, 1999.

[13] N. H. Gartner, C. J. Messer, and A. K. Rathi. *Monograph on traffic flow theory*. Federal Highway Administration, 1997.

[14] B. George and S. Shekhar. SP-TAG: a routing algorithm in non-stationary transportation networks. In *5th Proc. MobiQuitous 2008*, 2008.

[15] J. Gil Lee, J. Han, and K. Whang. Trajectory clustering: A partition-and-group framework. In *Proc. ACM SIGMOD*, pages 593–604, 2007.

[16] R. H. Güting and M. Schneider. *Moving Objects Databases*. Morgan Kaufmann, 2005.

[17] T. Hill. Conflations of probability distributions. *Trans. American Mathematical Society*, 363(6):3351–3372, 2011.

[18] T. P. Hill and J. Miller. How to combine independent data sets for the same quantity. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 21(3):033102, 2011.

[19] V. Kogan, I. Shimshoni, and D. Levi. Lane-level positioning with sparse visual cues. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 889–895. 2016.

[20] G. Laporte. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59(3):345–358, 1992.

[21] C. Lin, K. L. Choy, G. T. Ho, S. Chung, and H. Lam. Survey of green vehicle routing problem: Past and future trends. *Expert Systems with Applications*, 41(4):1118–1138, 2014.

[22] J. Lin. Divergence measures based on the Shannon entropy. *IEEE Trans. on Information theory*, 37(1):145–151, 1991.

[23] P. Newson and J. Krumm. Hidden Markov map matching through noise and sparseness. *Proc. ACM SIGSPATIAL GIS*, pages 336–343, 2009.

[24] J. Niedermayer, A. Züfle, T. Emrich, M. Renz, N. Mamoulis, L. Chen, and H.-P. Kriegel. Probabilistic nearest neighbor queries on uncertain moving object trajectories. *PVLDB*, 7(3):205–216, 2013.

[25] A. Pascale, F. Deflorio, M. Nicoli, B. D. Chiara, and M. Pedroli. Motorway speed pattern identification from floating vehicle data for freight applications. *Transportation Research Part C: Emerging Technologies*, 51:104 – 119, 2015.

[26] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

[27] M. a. Quddus, W. Y. Ochieng, and R. B. Noland. Current map-matching algorithms for transport applications: State-of-the art and future research directions. *Transportation Research Part C: Emerging Technologies*, 15:312–328, 2007.

[28] J. Rabe, M. Necker, and C. Stiller. Ego-lane estimation for lane-level navigation in urban scenarios. In *Intelligent Vehicles Symposium (IV), 2016 IEEE*, pages 896–901. IEEE, 2016.

[29] H. G. Seif and X. Hu. Autonomous driving in the icity-hd maps as a key challenge of the automotive industry. *Engineering*, 2(2):159–162, 2016.

[30] J. Shang, Y. Zheng, W. Tong, E. Chang, and Y. Yu. Inferring gas consumption and pollution emission of vehicles throughout a city. In *Proc. ACM SIGKDD*, pages 1027–1036. 2014.

[31] C. E. Shannon. A mathematical theory of communication. *Univ. of Illinois Press*, 1971.

[32] G. Trajcevski, A. Choudhary, O. Wolfson, Y. Li, and G. Li. Uncertain range queries for necklaces. In *IEEE MDM*, 2010.

[33] G. Trajcevski, O. Wolfson, K. Hinrichs, and S. Chamberlain. Managing uncertainty in moving objects databases. *ACM Trans. Database Syst.*, 29(3), 2004.

[34] F. van Diggelen. Update: GNSS accuracy: Lies, damn lies, and statistics. *GPS World*, 2007.

[35] A. Vu, A. Ramanandan, A. Chen, J. Farrell, and M. Barth. Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation. *IEEE Trans. Intelligent Transportation Systems*, 13(2):899–913, 2012.

[36] Y. Wang, Y. Zheng, and Y. Xue. Travel Time Estimation of a Path using Sparse Trajectories. *Proceeding of the 20th SIGKDD*, (5), 2014.

[37] C. Wenk, R. Salas, and D. Pfoser. Addressing the need for map-matching speed: Localizing global curve-matching algorithms. In *SSDBM*, 2006.

[38] P. A. Zandbergen. Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning. *Transactions in GIS*, 13(s1):5–25, 2009.

[39] B. Zhang, G. Trajcevski, and F. Liu. Clustering speed in multi-lane traffic networks. In *Proc ACM CIKM, pages 2045–2048*, 2016.

[40] B. Zhang, G. Trajcevski, and L. Liu. Towards fusing uncertain location data from heterogeneous sources. *GeoInformatica*, pages 1–34, 2015.

[41] T. Zhang, S. Arrigoni, M. Garozzo, D.-g. Yang, and F. Cheli. A lane-level road network model with global continuity. *Transportation Research Part C: Emerging Technologies*, 71:32–50, 2016.