

On Map-Matching Vehicle Tracking Data

Sotiris Brakatsoulas¹

Dieter Pfoser¹

Randall Salas²

Carola Wenk²

¹RA Computer Technology Institute
Akteou 11
GR-11851 Athens, Greece
{sbrakats, pfoser}@cti.gr

²Department of Computer Science
University of Texas at San Antonio
San Antonio, TX 78249-0667, USA
{rsalas, carola}@cs.utsa.edu

Abstract

Vehicle tracking data is an essential “raw” material for a broad range of applications such as traffic management and control, routing, and navigation. An important issue with this data is its accuracy. The method of sampling vehicular movement using GPS is affected by two error sources and consequently produces inaccurate trajectory data. To become useful, the data has to be related to the underlying road network by means of *map matching* algorithms. We present three such algorithms that consider especially the trajectory nature of the data rather than simply the current position as in the typical map-matching case. An incremental algorithm is proposed that matches consecutive portions of the trajectory to the road network, effectively trading accuracy for speed of computation. In contrast, the two global algorithms compare the entire trajectory to candidate paths in the road network. The algorithms are evaluated in terms of (i) their running time and (ii) the quality of their matching result. Two novel quality measures utilizing the Fréchet distance are introduced and subsequently used in an experimental evaluation to assess the quality of matching real tracking data to a road network.

1 Introduction

As roads become more and more congested, much research is conducted in the area of traffic estimation and

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the VLDB copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Very Large Data Base Endowment. To copy otherwise, or to republish, requires a fee and/or special permission from the Endowment.

**Proceedings of the 31st VLDB Conference,
Trondheim, Norway, 2005**

prediction systems (TREPS). TREPS use *traffic models* together with *sensor data* to assess the current and to predict the future traffic conditions in the road network. Currently, the data component consists of *traffic counts* (quantitative data) obtained by stationary sensors, typically loop detectors, which are deployed throughout the road network. In recent years, a new sensor technology is utilized to complement stationary sensor networks. *Floating car data* (FCD) refers to using data generated by one vehicle as a sample to assess the overall traffic conditions (“cork swimming in the river”). Typically this data comprises basic vehicle telemetry such as speed direction and most importantly the position of the vehicle. Recording the position of vehicles in time produces tracking data, of which, in connection with a road network, *travel time data* (qualitative data) is derived. Having large numbers of vehicles collecting such data for a given spatial area such as a city (e.g., taxis, public transport, utility vehicles, private vehicles, etc.) will create an accurate picture of the traffic condition in time and space [10]. Data management techniques for such FCD collections are presented in [6].

The tracking data is obtained by sampling the positions using typically GPS to produce data that in database terms is commonly referred to as trajectories. Unfortunately, this data is not precise due to the *measurement error* caused by the limited GPS accuracy, and the *sampling error* caused by the sampling rate [14]. A pre-processing step that matches the trajectories to the road network is needed. This technique is commonly referred to as *map matching*.

Most map-matching algorithms are tailored towards mapping *current positions* onto a vector representation of a road network. Onboard systems for vehicle navigation utilize besides continuous positioning also dead reckoning to minimize the positioning error and to produce accurate vehicle positions that can be easily matched to a road map. In the given context, the *entire trajectory* given as a sequence of historic position samples needs to be mapped. The fundamental difference in these two approaches is the error associated with the data. Whereas the data in the former case is

mostly affected by the measurement error, the latter case is mostly concerned with the sampling error.

We present three map-matching algorithms that map a trajectory onto a road network by matching geometries. The methods differ in the portion of the trajectory they consider for this task. The first approach employs an *incremental match* of the position samples by pursuing a local match of geometries, i.e., matching a portion of the trajectory onto a path in the road network by using a measure composed of distance and angles between the curves. This approach effectively trades accuracy for speed of computation. The second approach aims for a *global match* mapping the entire trajectory to a candidate curve in the road network. Two similarity measures are used, the Fréchet distance and the weak Fréchet distance, resulting in two different map-matching algorithms which guarantee to find a matching curve with optimal distance to the trajectory.

Assessing the quality of the matching result is challenging since for vehicle tracking data typically the “true” path of the vehicle in the road network is not known. Thus, in order to still be able to evaluate the quality of the matching result, a distance measure between the trajectory and the matched path in the road network is needed. We propose two quality measures: (i) the Fréchet distance and (ii) the average Fréchet distance. In an experimental evaluation, real tracking data is used to evaluate the matching results in a practical setting. The speed of computation is assessed through analytical cost formulae detailing the running times of the algorithms.

The Fréchet distance for two curves has been proposed by Fréchet [8], and Alt et al. [2] gave an algorithm for its computation. Although of practical interest, variations of the Fréchet distance such as summed or average Fréchet distance (c.f. Section 5.2) have not yet been considered in the literature. Computing the integral Fréchet distance is an open problem and addressed in this work. Alt et al. [1] give a map-matching algorithm based on the Fréchet distance. We will sketch this algorithm in Section 4.3. In addition, we propose to utilize the weak Fréchet distance which results in a map-matching algorithm with a reduced running time while at the same time producing identical matching results for trajectories.

Work in the area of map-matching vehicle positions exists towards augmenting GPS positioning with other methods such as dead reckoning to reduce the measurement error and to achieve better map-matching results (cf. [15]). Greenfeld [9] introduces a map-matching strategy based on distance and orientation that does not assert any further knowledge about the movement besides the position samples. This algorithm serves as the basic strategy for the algorithm introduced in Section 3. Civilis et al. [7] in their work on location update techniques for the tracking of users

in location-based services introduce a map-matching algorithm that is based on edge distance and direction similar to [9]. The tracking data itself is obtained by using an active sampling technique based on predicted and measured positions. By controlling the sampling rate, the sampling error can be kept minimal and the map-matching algorithm is presented with an optimal dataset. Yin and Wolfson [18] propose an algorithm based on a weighted graph representation of the road network in which the weights of each edge represent the distance of the edge to the trajectory. The matched trajectory in the road network is found by using a Dijkstra shortest-path algorithm for the weighted graph. This algorithm is based on a measure related to the average Fréchet distance, however no overall quality guarantee on the matched curve is given. The authors claim that the algorithm produces high quality matches, however details on the data set, such as type and size are missing. Finally, a general introduction to the map-matching problem as addressed in this work can be found in [5].

The outline of this paper is as follows. Section 2 discusses tracking data and introduces the error sources that affect it to derive requirements for map-matching algorithms. Sections 3 and 4 detail the incremental and the global map-matching algorithms, respectively. Section 5 defines the measures to assess the quality of the map-matching result. Section 6 shows the outcome of the experimental evaluation and, finally, Section 7 gives conclusions and directions for future research.

2 Motivation and Background

The objective of this work is to develop map-matching algorithms for vehicle tracking data that are used as a sensor data source to assess and predict the traffic condition in related applications. This section overall describes the properties of the tracking data with a focus on its accuracy to define requirements for map-matching algorithms.

2.1 Imprecise Trajectory Data

The tracking data can be modeled in terms of a trajectory, which is obtained by interpolating the position samples. Typically, linear interpolation is used as opposed to other methods such as polynomial splines. The sampled positions then become the endpoints of line segments of polylines, and the movement of an object is represented by an entire polyline in 3D space [14]. The positioning technology of choice for vehicle tracking is typically GPS. Its associated *measurement error* in connection with the *sampling rate* require us to match position samples to the road network.

2.1.1 Measurement Error

Two assumptions are generally made about the accuracy of GPS. First, the error distribution, i.e., the error

in each of the three dimensions and the error in time, is assumed to be *Gaussian*. Second, we can assume that the horizontal error distribution, i.e., the distribution in the x-y plane, is circular [17].

The error in a positional GPS measurement can be described by the probability function following a bivariate normal distribution [11]. The probability function is composed of two normal distributions in the two respective spatial dimensions. The mean of the distribution is the origin of the coordinate system. Fig. 1 visualizes the error distribution. In addition to the mean, the standard deviation, σ , is the characteristic parameter stated when referring to GPS accuracy, e.g., 5m GPS accuracy refers to σ . Within the range of $\pm\sigma$ of the mean, in a bivariate normal distribution, 39.35% of the probability is concentrated.

Although the GPS error can be substantial in certain situations (shadowed and reflected signals), with the use of augmenting the GPS signal (WAAS, EGNOS) the typical error is in the range of 8m to 2m.

The following section discusses the sampling error, which in the case of vehicle tracking data is by two magnitudes larger than the measurement error and consequently has a larger impact on the algorithmic design of a map-matching algorithm.

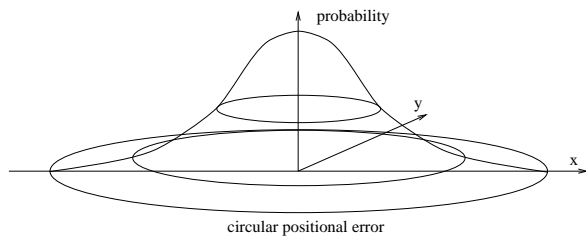


Figure 1: Measurement error for GPS.

2.1.2 Sampling Error

The uncertainty of the representation of an object's movement is affected by the frequency with which position samples are taken, the *sampling rate*. To illustrate the impact of the sampling rate on the imprecision of the interpolated trajectory data, consider sampling the position of a vehicle every 30s, which is a typical sampling rate used in vehicle tracking applications. At a speed of 50km/h, the traveled distance between position samples is as much as 417m!

Not measuring the positions in-between two consecutive position samples, the best we can do is to *limit the possibilities of where the moving object could have been*. The trajectory can be constrained by what we know about the object's actual movement.

The authors in [14] show that the sampling error between two positions, P_1 and P_2 in the time interval $[t_1, t_2]$ and a given *maximum speed*, v_m , for a time t_x with $t_1 < t_x < t_2$ is bound by a lens-shaped probability distribution. Computing the sampling error for

various instances of t_x shows that a possible trajectory between P_1 and P_2 is overall bound by an error ellipse (cf. Fig. 2). The foci of the ellipse are the sampled positions P_1 and P_2 and the eccentricity $2c$ is the Euclidean distance between P_1 and P_2 . The length of the semi-major axis, $2a$, is the maximum distance the object can travel. If the sampling rate r is given in seconds and the velocity is v km/h then $2a = 5/18 \cdot vr$. The "thickness" of the ellipse, $2b$, is determined by the equation $b^2 = a^2 - c^2 = 25/36^2 \cdot v^2 r^2 - c^2$. In simple words, the faster the object travels and the closer its path to that of a linear trajectory, the "thinner" the ellipse. In extreme cases, the ellipse degrades to a *line*, or even to a *point* in case the object stopped.

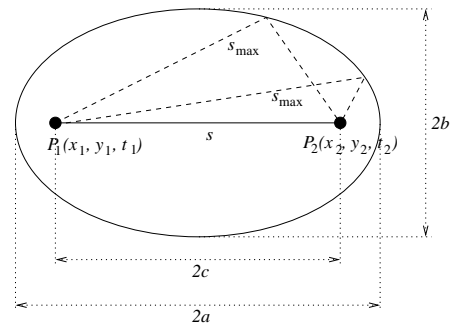


Figure 2: Error ellipse.

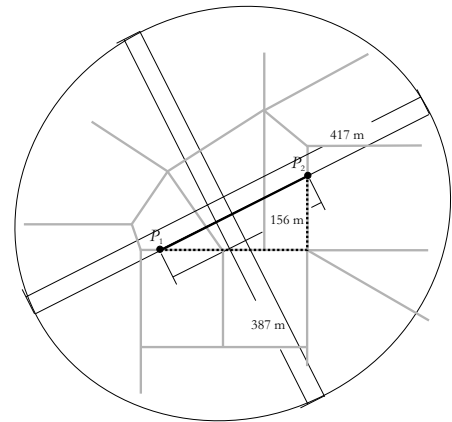


Figure 3: Error ellipse.

Considering again the initial example, Figure 3 gives its sampling error scenario. On a road network (gray lines), a vehicle travels along the dotted road network edge from P_1 to P_2 at a typical speed of 50km/h. Since it has to stop at the intersection, its average speed is 25km/h. Between P_1 and P_2 , the traveled distance along the road network is 208m (length of dotted path) and assuming P_1 is 140m and P_2 68m from the intersection, the Euclidean distance is 156m. The resulting error ellipse has a major axis of $2a = 5/18 \cdot 50\text{km/h} \cdot 30\text{s} = 417\text{m}$ and an eccen-

tricity of $2c = 156m$. The thickness of the ellipse is $2b = 387m$. Translated to a map-matching task, one would have to consider all the network edges contained in this error ellipse. Additional knowledge such as the number of intersections on a path reduce the possibilities, but several possible alternatives for mapping the trajectory onto the road network remain.

Figure 4 gives an example of a vehicle trajectory composed of GPS measurements (asterixes connected by line segments). Typically the GPS measurements do not lie on the road network (measurement error) and neither can the connecting line segments easily be matched to edges of the road network (sampling error).

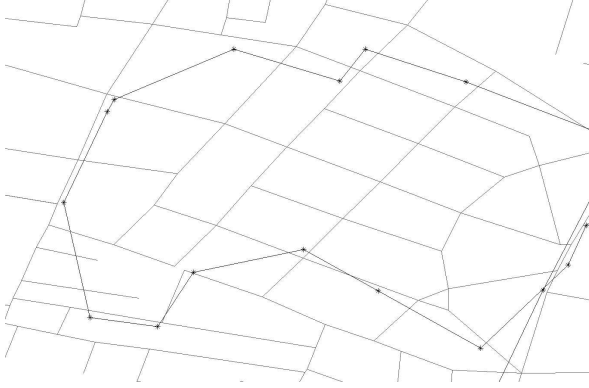


Figure 4: GPS error in vehicle tracking data.

2.2 Trajectories and Travel Times

Besides the accuracy of the trajectory data, the final use of the map-matching result affects the design of a respective algorithm. In utilizing vehicle tracking data as a data source for traffic estimation and prediction, travel times are derived from the map-matched trajectory data. Modeled as a graph, edges and vertices constitute a road network. Figure 5 presents the corresponding conceptual schema by using a common Entity-Relationship representation. Edges are assigned travel times as derived from tracking data, i.e., given a specific trajectory, how long did it take to traverse the edge in question. The travel times are recorded by means of absolute timestamps, “time1” and “time2” for entering and leaving the edge, respectively, and the direction in which the edge was traversed. Consequently, a requirement for a map-matching algorithm is that achieving the best match for the *entire* trajectory is not necessary, but one only needs to be able to identify portions of “bad” matches to derive accurate travel times.

3 Incremental Map-Matching Algorithm

An important objective when dealing with massive amounts of incoming tracking data is to create a fast map-matching algorithm. Using a greedy strategy

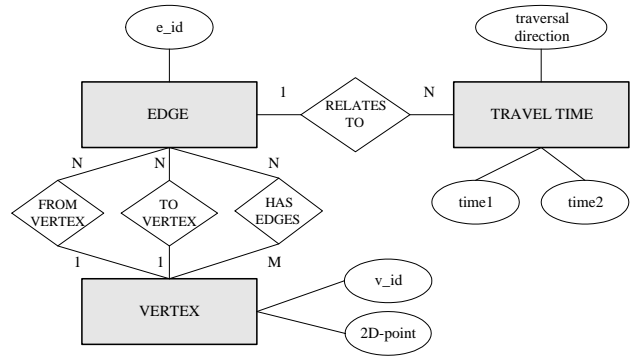


Figure 5: Data model excerpt.

based on local spatial characteristics, position samples comprising the trajectory are matched sequentially by in each case comparing the geometry of a portion of the trajectory to selected paths in the road network.

3.1 The Basic Algorithm

Given a series of position samples representing a vehicle trajectory, the map-matching algorithm pursues a position-by-position sample and edge-by-edge strategy. To match a position p_1 to a road network edge, given that its previous position p_{i-1} has already been matched, the algorithm proceeds as follows (c.f. Figure 6). First, the candidate edges to be matched to the current position are identified as the set of the incident edges “exiting” the last matched edge (including also the matched edge itself). In Figure 6, these edges are labeled c_1, c_2 and c_3 , with c_3 being the edge matched to p_{i-1} .

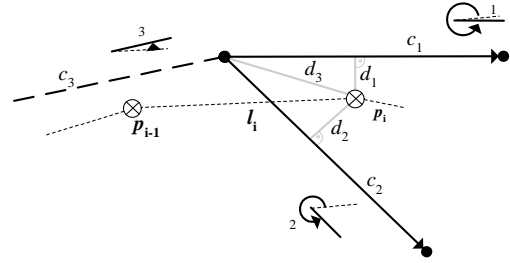


Figure 6: Incremental map-matching example.

Two similarity measures are used to evaluate the candidate edges [9].

The measure s_d reflecting the *distance* from the position sample to the various edges is computed based on the weighted line segment distance¹, d , of p_i from

¹The *line segment distance* of a point to a line is either the perpendicular line distance if the projection of the point onto the line segment is between its endpoints, or, otherwise, it is the distance of the point to the closest endpoint of the line segment.

each candidate, c_j , using the scaling factors μ_d and n_d as

$$s_d(p_i, c_j) = \mu_d - a \cdot d(p_i, c_j)^{n_d}.$$

The measure s_α reflects the *orientation* of the trajectory with respect to the candidate edge. It is computed based on the angle difference $\alpha_{i,j}$ between the directed candidate edge c_j and the directed line segment $l_i = \overrightarrow{p_{i-1}, p_i}$, using the scaling factors μ_α and n_α as

$$s_\alpha(p_i, c_j) = \mu_\alpha \cdot \cos(\alpha_{i,j})^{n_\alpha}.$$

The scaling factors $\mu_{[d|\alpha]}$ and $n_{[d|\alpha]}$ represent the maximum score and a power parameter, respectively. Choosing a higher μ_d compared to μ_α means that distance weighs more than orientation. The power parameter determines the rate of decrease for the respective weight with an increasing line segment distance or angle difference. The use of the cosine further implies that with an increasing angle difference the score of s_α decreases and with angle differences $90 < \alpha < 270$ and the choice of an odd number for the power n_α and a positive constant μ_α , s_α even becomes negative. For the specific dataset used in this work (cf. Section 6.2.1), we empirically established the following parameter settings $\mu_d = 10$, $a = 0.17$, $n_d = 1.4$, and $\mu_\alpha = 10$, $n_\alpha = 4$.

The combined similarity measure is computed as the sum of the individual scores, i.e.,

$$s = s_\alpha + s_d.$$

The higher the score of this measure, the better is the match.

Depending on the type of projection/match of p_i to c_j , i.e., (i) its projection is between the end points of c_j , or, (ii) it is projected onto the end points of the line segment, the algorithm does, or does not advance to the next position sample. Following the example of Figure 7, after matching p_1 to edge e_1 , the algorithm advances to p_2 (case (i)) and matches it also to e_1 . Advancing to p_3 , it tries to match this point to e_2 and since this projection reflects case (ii) it does not advance to the next position sample but finally matches e_3 to p_3 . The edge e_2 is recorded as a traversed edge. In Figure 7, the mapped position samples are drawn as gray circled crosses.

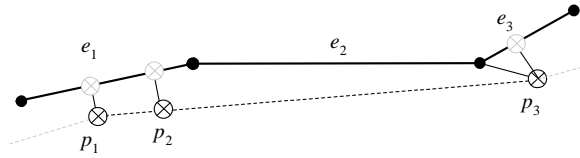


Figure 7: Matching example: advancing position samples and edges, and matching result.

3.2 Introducing Local Look-Ahead

To improve this simple algorithm, a recursive “look-ahead” policy has been adopted. Recursively, for each candidate edge c_j , the score of the best candidate among its “exiting” edges $c_{j,k}$ is calculated. This strategy aims at making a more global matching decision by exploring alternative branches rather than simple edges. Finally, only one choice is materialized in the matching result. The number of edges in the look-ahead is fixed. We established empirically that a look-ahead of 4 edges (the candidate edge plus three more edges) is optimal in terms of matching quality and time of computation.

Figure 8 shows a simple example with a look-ahead of 2 edges. With c_2 being a candidate edge, $c_{2,1}$ is the best candidate for matching p_{i+1} if we would match p_i to c_2 . For c_1 as a candidate edge, c_1 is also the best candidate for matching p_{i+1} , if we would match p_i to c_1 .

The final scores for matching p_i to the edges c_j (c_1, c_2 , and c_3 in our example) are computed as the sum of the scores for each “best” subpath as

$$s(p_i, c_j) = \sum_{k,l=0}^{depth} s(p_{i+l}, c_{j,k}).$$

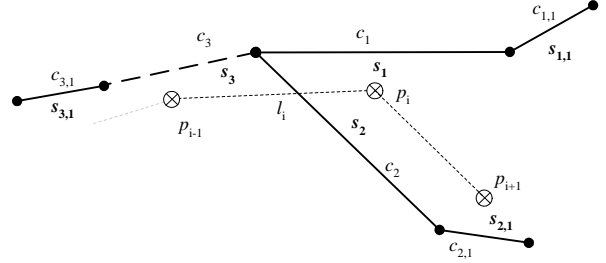


Figure 8: Incremental map-matching example with look ahead.

3.2.1 Initialization

To apply the above matching strategy, the algorithm needs to be *initialized* by mapping the first position sample p_0 . To be able to use the above algorithm, we have to find for the directed line segment $l_0 = \overrightarrow{p_0, p_1}$ an initial set of candidate edges E_0 . Using a simple approach, E_0 contains all edges that are within a threshold distance to p_0 . The threshold distance is related to the GPS error and was for our specific case set to 100m. All edges in E_0 are then evaluated with respect to l_0 to determine a match for p_0 .

3.3 Performance Considerations

To match a trajectory that consists of n position samples, the algorithm has to evaluate for each sample a

finite number of adjacent road network edges a with a maximum look-ahead of l edges. Consequently, its running time is $O(na^{l+1})$. Given that, both, a and l are essentially constants, the algorithm effectively runs in $O(n)$ time. The initialization cost is determined by finding the set of closest edges to a position. Using an adjacency list (cf. Figure 5), this search can be achieved in $O(\log v + w)$, where v corresponds to the number of vertices in the network and w to the size of the result set [12]. What can be assumed is that the actual map-matching time $O(n)$ dominates the initialization time of $O(\log v + w)$.

For a disk-based algorithm, the running time will depend highly on the representation and the storage of the road network. The performance of such an algorithm can be improved if the road network is stored by means of tiles, i.e., the road network is spatially subdivided into rectangular tiles, and edges are not retrieved individually but rather as collections belonging to the same tile.

4 Global Map-Matching Algorithm

A global map-matching strategy tries to find a curve in the road network (modeled as a graph embedded in the plane with straight-line edges) that is as close as possible to the vehicle trajectory. In our approach, we minimize over all possible curves in the road network. For the comparison between two curves, we employ the *Fréchet distance* or the *weak Fréchet distance*. Our aim is to design algorithms which utilize these distance measures to give a *quality guarantee* on the computed result.

Alt et al. [1] have designed an algorithm solving the global map-matching task using the Fréchet distance in $O(mn \log^2 mn)$ time, where m is the total number of vertices and edges in the road network and n , as previously, is the number of position samples of the vehicle trajectory. In the following sections we will give basic definitions, sketch this algorithm, and we will give an algorithm for the global map-matching task based on the *weak Fréchet distance* which runs in $O(mn \log mn)$ time.

4.1 Fréchet Distance

The Fréchet distance for two curves has been proposed by Fréchet [8]. A popular illustration of the Fréchet distance is the following: Suppose a person is walking his dog, the person is walking on the one curve and the dog on the other. Both are allowed to control their speed but they are not allowed to go backwards. Then the Fréchet distance of the curves is the minimal length of a leash that is necessary for both to walk the curves from beginning to end.

Formally, the Fréchet distance between two curves $f, g: [0, 1] \rightarrow \mathbb{R}^2$ is defined as

$$\delta_F(f, g) := \inf_{\alpha, \beta: [0, 1] \rightarrow [0, 1]} \max_{t \in [0, 1]} \|f(\alpha(t)) - g(\beta(t))\|,$$

where α and β range over continuous and non-decreasing reparametrizations with $\alpha(0) = \beta(0) = 0$ and $\alpha(1) = \beta(1) = 1$ only. If we drop the requirement on α and β to be non-decreasing, we obtain a distance measure $\tilde{\delta}_F(f, g)$ that is called the *weak Fréchet distance* between f and g .

The Fréchet distance as well as the weak Fréchet distance are especially well-suited for the comparison of trajectories since they take the continuity of the curves into account, c.f. Section 5.1. Notice that $\tilde{\delta}_F(f, g) \leq \delta_F(f, g)$, since the weak Fréchet distance minimizes over more reparametrizations than the Fréchet distance.

4.2 Freespace Diagram and Freespace Surface

We first consider the decision variant of the global map-matching problem: For a fixed $\varepsilon > 0$ decide whether there exists a curve in the road network with distance at most ε to the vehicle trajectory. The actual minimization problem can then be solved by either applying parametric search (which adds a log-factor to the runtime), or binary search (which is more feasible to implement in practice). If not stated otherwise let $\varepsilon > 0$ be given.

For two curves $f, g: [0, 1] \rightarrow \mathbb{R}^2$ we call the set $F_\varepsilon(f, g) := \{(s, t) \in [0, 1]^2 \mid \|f(s) - g(t)\| \leq \varepsilon\}$ the *free space* of f and g . Here $\left\| \begin{pmatrix} x \\ y \end{pmatrix} \right\| = \sqrt{x^2 + y^2}$ denotes the L_2 -norm. The partition of $[0, 1]^2$ into regions belonging or not belonging to $F_\varepsilon(f, g)$ is called the *free space diagram*. Figure 9 shows polygonal curves f, g , a distance ε , and the corresponding free space diagram with the free space $F_\varepsilon = F_\varepsilon(f, g)$ indicated in white. A monotone curve from the lower left corner to the upper right corner is drawn in the free space. This illustration is taken from [2]. The free space of two line segments is an ellipse [2] and the free space diagram of two polygonal curves of m and n segments is composed of mn segment-segment free space diagrams. In [2] it has been shown that $\delta_F(f, g) \leq \varepsilon$ if and only if there exists a curve within $F_\varepsilon(f, g)$ from the lower left corner $(0, 0)$ to the upper right corner $(1, 1)$, which is monotone in both coordinates. Observe that the monotone curve in $F_\varepsilon(f, g)$ from the lower left corner to the upper right corner as a continuous mapping from $[0, 1]$ to $[0, 1]^2$ directly gives continuous non-decreasing reparametrizations α and β . Similarly, $\tilde{\delta}_F(f, g) \leq \varepsilon$ if and only if there exists a curve within $F_\varepsilon(f, g)$ from the lower left corner to the upper right corner, which is not necessarily monotone.

The definition of the free space between two curves generalizes to the free space between an embedded graph and a curve as follows: The free space of the road network and the trajectory is the union of all free spaces of all straight-line edges of the road network with the vehicle trajectory. Notice that the free space of a vertex v with the vehicle trajectory is a one-dimensional free space, and the individual free spaces

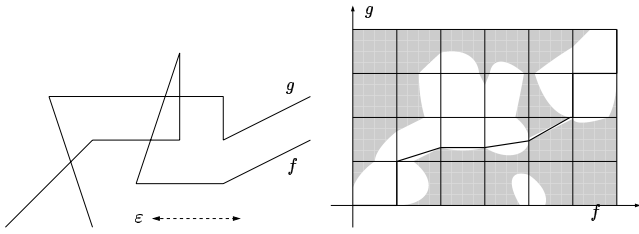


Figure 9: Free space diagram for two polygonal curves f and g .

of all edges incident to v with the trajectory share the one-dimensional free space at v . Thus we can glue together the two-dimensional free space diagrams along the one-dimensional free space they have in common, according to the adjacency information in the road network. We call this topological structure, which is the union of all edge-trajectory free space diagrams, the *free space surface* of the road network and the trajectory. Figure 10 gives an example road network (left) and its corresponding free space surface and a vehicle trajectory consisting of five position samples (right). The vehicle trajectory is not shown explicitly but implicitly by the white free space area. An example path in the free space from lower left to upper right is drawn dashed.

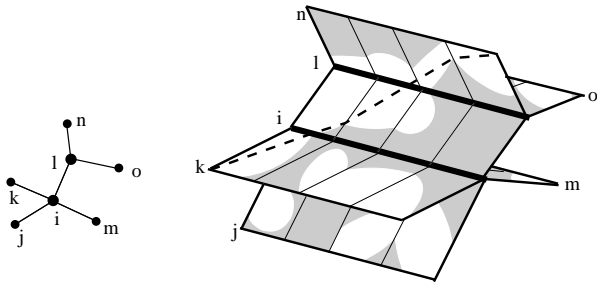


Figure 10: A road network (left) and corresponding free space surface (right).

4.3 Using the Fréchet Distance

The decision problem for the Fréchet distance between two curves can be solved by computing a monotone curve from the lower left corner to the upper right corner in the free space. This can be done using a dynamic programming approach in $O(mn)$ time [2].

Alt et al. [1] generalized this approach to finding a monotone path in the free space surface from a lower left corner of some individual edge-trajectory free space diagram to an upper right corner of some other individual edge-trajectory free space diagram. The algorithm conceptually sweeps a line from left to right (in direction of the trajectory) over all free spaces at the same time while maintaining the points on the

sweep line that are reachable by some monotone path in the free space from some lower left corner. It then updates this reachability information Dijkstra-style while advancing the sweep line. Interestingly, the algorithm runs in $O(mn \log mn)$ time, which is only a log-factor slower than the algorithm of [2], although it accomplishes the seemingly more complicated task of comparing the trajectory to all possible curves in the road network.

4.4 Using the Weak Fréchet Distance

The decision problem for the weak Fréchet distance between two curves can be solved by testing if there exists any path in the free space of the two curves from the lower left corner to the upper right corner. This can be done using any graph traversal algorithm such as depth-first search in $O(mn)$ time.

We generalize this approach to the global map-matching problem by applying depth first search to the free space surface. We initialize the search with all white lower left corners of individual edge-trajectory free spaces, and stop the search if we found some upper right white corner. Since the free space surface consists of mn edge-segment cells, this algorithm runs in $O(mn)$ time, which is a log-factor faster than the algorithm based on the Fréchet distance. Applying parametric search for optimization, in the same way as in [2, 1] adds an additional log-factor to the runtime for a total of $O(mn \log mn)$ to solve the optimization problem.

5 Quality Measures

To determine the quality of the map-matching results, we need to introduce a quality measure that evaluates how closely the matched trajectory resembles the original one. We will utilize *distance measures*, which have smaller values for more similar curves, as opposed to *similarity measures* (c.f. Section 3.1), which have larger values for more similar curves.

5.1 A Suitable Distance Measure

There are several distance measures for two curves available in the literature, such as the Hausdorff distance, the Fréchet distance, the weak Fréchet distance, the turn-angle distance, etc. See [3] for an overview of distance measures for various kinds of shapes. Although the Hausdorff distance is a widely used distance measure for shapes in general, it is not suitable for the case of curves since it does not take the continuity of the curves into account; it assigns to every point on one curve the closest point on the other and maximizes over all these distances. The Hausdorff distance is the maximum of the two values obtained by considering each of the two curves as the first curve. The Fréchet distance on the other hand takes the continuity into account in that it only allows monotone and

continuous assignments between points on the curves. In fact, the Hausdorff distance can be arbitrarily small for two curves that should intuitively have a large distance; see Figure 11 for two curves with small Hausdorff and large Fréchet distance and for two curves with a small weak Fréchet distance and a large Fréchet distance.

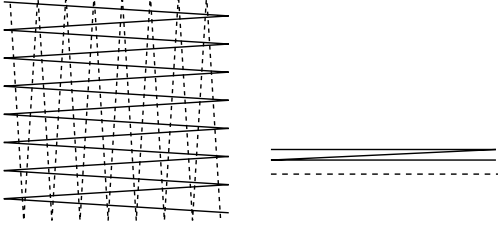


Figure 11: Two curves: with a small Hausdorff distance and a large Fréchet distance (left) and with a small weak Fréchet distance and a large Fréchet distance (right).

The choice between weak Fréchet distance and Fréchet distance depends on the application, whether the continuous reparametrizations under consideration need to be monotone or not. In our case, “backing up” on a road in between two intersections (i.e., on an edge in the road map) seems unlikely, however backing up on a vehicle trajectory, although unlikely, is conceivable if the sampling was too coarse and hence skipped a zig-zag part of the road network (see Figure 11 right, with the solid curve as the road network). Although we could define a *semi-weak* Fréchet distance which considers monotone reparametrizations in only one coordinate, we will nevertheless model our situation with the Fréchet distance. Our experiments in Section 6.2 strongly support this model.

5.2 Average Fréchet Distance

The Fréchet distance takes the maximum over a set of distances and thus it is strongly affected by outliers. As discussed before, the vehicle trajectory is prone to errors and outliers and it would therefore be desirable to consider a distance measure which averages over certain distances instead of taking the maximum. Let us therefore define the *integral Fréchet distance* as

$$\delta_{\mathbb{F}}^{\text{int}}(f, g) = \inf_{\alpha, \beta: [0,1] \rightarrow [0,1]} \int_{(\alpha, \beta)} \|f(\alpha) - g(\beta)\| dt,$$

where the integral denotes the path integral over the curve $t \mapsto (\alpha(t), \beta(t))^T$, dt denotes the path element along this curve, and α and β range over differentiable and non-decreasing reparametrizations with $\alpha(0) = \beta(0) = 0$ and $\alpha(1) = \beta(1) = 1$ only. The path integral can be rewritten as

$$\int_0^1 \|f(\alpha(t)) - g(\beta(t))\| \left\| \begin{pmatrix} \alpha'(t) \\ \beta'(t) \end{pmatrix} \right\| dt.$$

Note that the path integral correctly integrates over all possible values of $\|f(\alpha(t)) - g(\beta(t))\|$ for fixed reparametrizations α and β , as desired by our application. However, it adds the differentiability restriction to α and β . Furthermore, in order to compare the integral Fréchet distance between several pairs of curves one needs to normalize the distance by dividing $\delta_{\mathbb{F}}^{\text{int}}(f, g)$ by the arclength of the optimizing curve (α, β) which is

$$\int_0^1 \left\| \begin{pmatrix} \alpha'(t) \\ \beta'(t) \end{pmatrix} \right\| dt.$$

Unfortunately there is no algorithm known that computes the integral Fréchet distance. However, we can approximate the integral Fréchet distance by sampling the curves and approximating the integral by a sum. Let $0 = i_1 < i_2 < \dots < i_M = 1$ be the sample positions for f and let $0 = j_1 < j_2 < \dots < j_N = 1$ be the sample positions for g . Then the *summed Fréchet distance* $\delta_{\mathbb{F}}^{\text{sum}}(f, g)$ is defined as

$$\min_{P=\binom{\alpha}{\beta}} \sum_{k=2}^{|P|} \|f(i_{\alpha(k)}) - g(j_{\beta(k)})\| \left\| \begin{pmatrix} i_{\alpha(k)} - i_{\alpha(k-1)} \\ j_{\beta(k)} - j_{\beta(k-1)} \end{pmatrix} \right\|,$$

where P ranges over all non-decreasing paths on $\{i_1, i_2, \dots, i_M\} \times \{j_1, j_2, \dots, j_N\}$ that start in $(i_1, j_1) = (0, 0)$ and end in $(i_M, j_N) = (1, 1)$. This definition is similar to the *edit distance* for strings and the *dynamic time warping (DTW)* in speech recognition, see [16] for an overview. Similar to the computation of the edit distance and the dynamic time warping, the summed Fréchet distance can be expressed as a recurrence in a straight-forward manner which allows a dynamic programming solution that runs in $O(MN)$ time. Notice that, different from the standard edit distance, the distance $\|f(i_{\alpha(k)}) - g(j_{\beta(k)})\|$ is weighed with the length of the segment of the reparametrization path.

The sampling we use in our implementation is almost uniform; we keep all original curve vertices. For a given arclength parameter λ we split every edge of the original curve with (arc-)length greater than λ into segments of equal length which lies between $\lambda(1 - \frac{1}{2k})$ and $\lambda(1 + \frac{1}{2k})$, where k is the length of the original curve edge divided by λ and rounded to the nearest integer. In our implementation we use $\lambda = 2$ meters.

Afterwards we normalize $\delta_{\mathbb{F}}^{\text{sum}}(f, g)$ by dividing by the arclength of the optimizing path

$$\sum_{k=2}^{|P|} \left\| \begin{pmatrix} i_{\alpha(k)} - i_{\alpha(k-1)} \\ j_{\beta(k)} - j_{\beta(k-1)} \end{pmatrix} \right\|.$$

We call the resulting distance measure the *average Fréchet distance*.

Notice that ideally it would be better to introduce this kind of normalization within the minimization, instead of normalizing after an optimal path has been

found. For the edit distance there have been discussions on this topic as well as algorithms for certain cases, see [13, 4]. In our case however, the cubic algorithm of [13] cannot be applied due to the nonuniform length of the sampled segments; it would require sampling at a strictly uniform rate, which in turn would not allow to keep all original trajectory vertices.

6 Performance Study

The objective of the performance study is to evaluate the three map-matching algorithms in terms of (i) their running times and (ii) the quality of their respective matching results.

6.1 Running Time

Table 1 gives an overview of the running times for the various algorithms as derived in Sections 3.3, 4.3 and 4.4, where n is the number of position samples of a trajectory, k is the number of edges, and m is the total number of edges and vertices in the road network. Note that the additional *log*-factor in the running times of the global map-matching algorithms, compared to the decision variants of Sections 4.3, 4.4, is introduced by the optimization using parametric search.

Incremental algorithm	Global algorithms	
	Fréchet dist.	Weak Fréchet dist.
$O(n)$	$O(mn \log^2 mn)$	$O(mn \log mn)$

Table 1: Performance comparison in terms of the asymptotic running time.

What can be readily observed is that the incremental algorithm is expected to run much faster than the global algorithms. The incremental map-matching algorithm was implemented in Java, and the global algorithms were implemented in C++. Subject to future work is to provide a common implementation platform to verify and to refine the asymptotic running times as presented in Table 1 by means of an empirical evaluation and to establish practical performance characteristics.

6.2 Empirical Evaluation

To compare the three algorithms, map-matching results for real tracking data were evaluated using the quality measures defined in Section 5.

6.2.1 The Data

Real tracking data was used in the experiment. It was obtained through GPS tracking with a sampling rate of 30 seconds. The dataset consists of 45 vehicle

trajectories consisting of a total of 4177 edges². The smallest and the largest trajectory consist of 57 and 148 edges, respectively. The road network consists of 14356 edges and represents a portion of the road network of the city of Athens, Greece. The road network is visualized in Figure 12.



Figure 12: Athens, Greece road network portion.

6.2.2 Map-Matching Quality

Interestingly, for each of the 45 trajectories both global matching algorithms (based either on the Fréchet distance or the weak Fréchet distance) computed the same results and the exact same distance (using binary search up to a precision of $0.5m$). This means that for each of our 45 trajectories, if there exists any path in the free space surface then it is already monotone, so the weak Fréchet distance actually equals the Fréchet distance. This suggests that situations like the one in Figure 11 right do not seem to occur in our data set since the GPS sampling rate was chosen high enough to prevent these situations. In generalizing this observation and pending further experiments, it seems to suffice for a global map-matching strategy to run the faster weak Fréchet distance-based algorithm. Since the matching results were identical, we, in the following, only refer to one global map-matching algorithm.

Map-matching results for all 45 trajectories were computed using the incremental and the global map-matching algorithm. All 90 matching results were evaluated by means of (i) the Fréchet distance (Figure 13) and (ii) the average Fréchet distance (Figure 14). The results in both charts are sorted according to the Fréchet distances of the global matching results, e.g., the first data point in Figures 13 and 14, respectively refers to the same matching results.

The evaluation in terms of the Fréchet distance clearly shows the global matching algorithm to pro-

²The vehicle tracking data was supplied by Emphasis Telematics, a co-operating telematics company and fleet management service provider.

duce far better results than the incremental algorithm. However, using the average Fréchet distance as a quality measure, the advantage of the global matching algorithm over the incremental one is reduced.

This difference is due to the fact that the Fréchet distance measure only considers the largest distance between the two curves. In some cases, the incremental algorithm does not find a good match for the first positions of a trajectory (initialization phase) due to a lack of knowledge about the initial trajectory geometry. The Fréchet distance considers only the worst case, which is very often this initial match. Considering the average Fréchet distance, such outliers have a smaller effect on the quality measure.

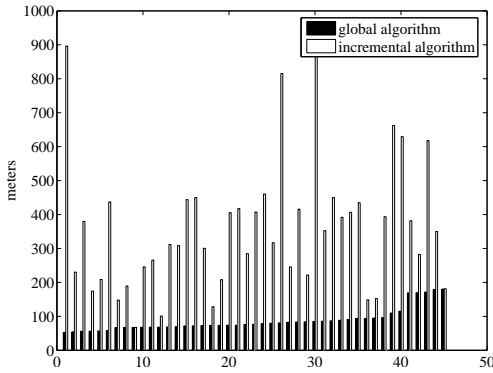


Figure 13: *Fréchet* distances between result curves and trajectory.

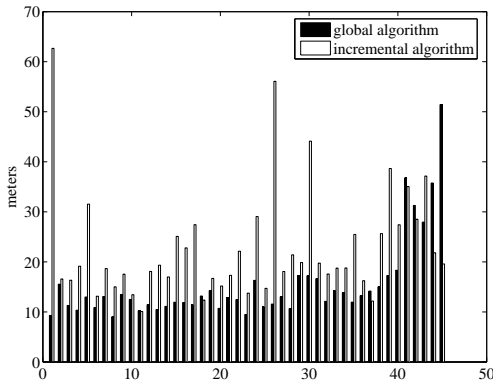


Figure 14: *Average Fréchet* distances between result curves and trajectory.

Figures 15 and 16 summarize the evaluation results by showing the average distance, its standard deviation, as well as the maximum and minimum distances for the Fréchet distance and the average Fréchet distance measure, respectively. Again, what can be observed is that by using the stricter quality measure of the Fréchet distance, the quality of the matching results produced by the global algorithm is even in terms

of summarized measures by far superior over that of the incremental algorithm. However, when evaluated with the average Fréchet distance, the quality of the map-matching results is comparable.

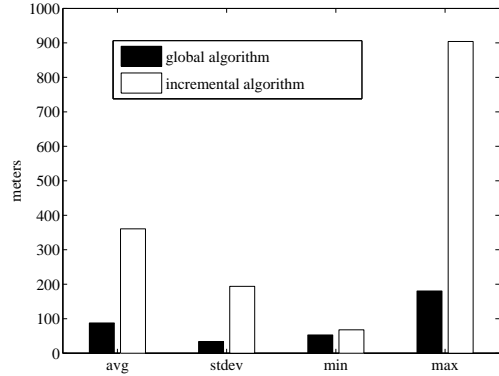


Figure 15: Summary of quality measure: *Fréchet* distances.

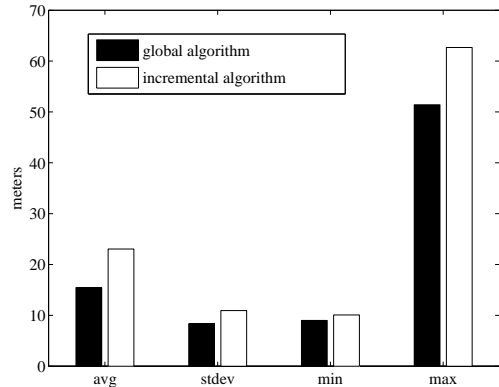


Figure 16: Summary of quality measure: *Average Fréchet* distances.

6.2.3 Examples

As a complement to the quality measures, Figures 17 and 18 present a visualization of two map-matching examples. A vehicle trajectory composed of GPS measurements (asterixes connected by line segments) is matched to a road network (gray grid) by means of the incremental algorithm (thick gray line) and the global algorithm (thinner black curve).

Figure 17 gives an example of a trajectory for which the two map-matching algorithms produced identical results. The Fréchet distance and the average Fréchet distance were 67.4 and 10.22 for the global algorithm and 67.8 and 10.08 for the incremental algorithm, respectively.

Figure 18 gives the map-matching results of the trajectory presented in Figure 4 as computed by the in-

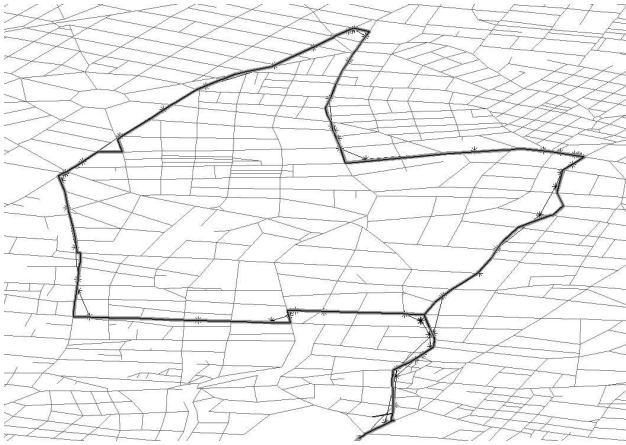


Figure 17: Identical map-matching results.

cremental algorithm (thin black curve) and the global algorithm (thicker gray line). The quality measures, the Fréchet distance and the average Fréchet distance, were 83.8 and 17.26 for the global algorithm and 221.7 and 19.87 for the incremental algorithm, respectively. Marked by two circular areas in this example are two typical map-matching problems. Area 1 illustrates the case in which the look ahead of the incremental algorithm is not large enough to detect the missing edge in the road network. Since the algorithm does not support backtracking, it produces a bad match until “closing in” again on the trajectory. The global algorithm produces a better match since it has a “global knowledge” of the road network and “detects” the gap.

Area 2 illustrates a case in which the two algorithms produce different matching results for an ambiguous trajectory. Both matching results seem acceptable by visual inspection, and certainly fulfill the constraints posed by the error ellipse of Section 2.1.

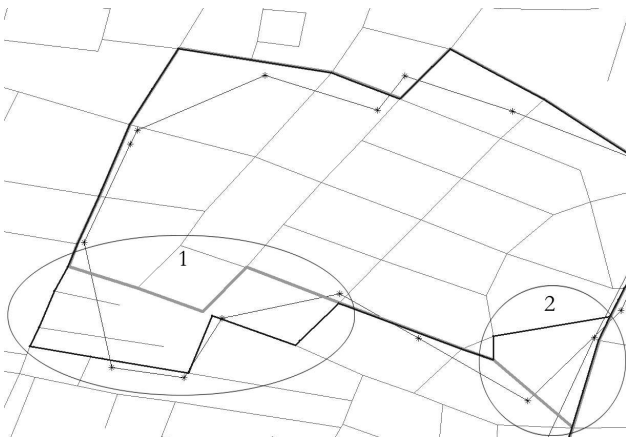


Figure 18: Typical map-matching issues.

Overall, tracking data exhibits two major particularities that complicate the map-matching process.

One, the sequences of position samples representing vehicle trajectories have gaps, i.e., large temporal and spatial differences between consecutive position samples (e.g., the GPS device was switched off). Two, the road network might have changed, i.e., the tracking data corresponds to edges in the road network that do not exist anymore. In both cases, the incremental algorithm stops and is re-initialized with the position sample that caused it to terminate. The global algorithm does not detect these conditions and tries to find a match for the entire trajectory.

6.3 Summary

The evaluation of the map-matching algorithms can be summarized as follows. While the global map-matching algorithm produces better matching results, the incremental algorithm produces results of lower quality faster. The quality measure that is better suited to compare the matching result also by considering the underlying motivation for this work, travel time computation, is the average Fréchet distance. It gives an estimate of the overall matching quality. The Fréchet distance is a stricter measure determining whether a good overall match was found.

7 Conclusions and Future Research

Vehicle tracking data constitutes an important resource for the application complex dealing with traffic assessment and prediction. Its utilization depends heavily on increasing the accuracy of the data and consequently relating it to the underlying road network by means of *map matching* algorithms.

We present three novel algorithms that exploit the trajectory nature of the tracking data, i.e., considering the entire path of a vehicle as opposed to its current position. The incremental map-matching algorithm employs a greedy strategy of sequentially matching portions of the trajectory to the road network. A local look ahead evaluates several alternate paths of fixed extents to introduce “globality” to the greedy strategy. The global map-matching algorithms find a curve in the road network that is as close as possible to the given trajectory. The underlying distance measure, in our case the Fréchet distance and the weak Fréchet distance, also serves as a quality guarantee for the computed result. As for the choice between Fréchet and weak Fréchet distance, the experiments showed that the two distance measures are equal in terms of the map-matching result they produced. The map-matching results were evaluated in terms of (i) their running time and (ii) the quality of their results. Comparing the asymptotic running times revealed that the incremental algorithm has a significant performance advantage over the global algorithms. However, the global algorithms were found to produce better matching results.

The directions for future work are as follows. The requirements for the map-matching algorithms are to produce travel times rather than perfect matching results. Thus, although the global algorithm produces good results, due to limitations in the tracking data and the road network, the matching results need to be evaluated to discard portions of “bad” matches (outliers). A common implementation of the three algorithms is needed to provide an empirical evaluation of the running times. The running time of the global map-matching algorithm can be improved by reducing the memory usage when only considering the free space within the error ellipse to store only the actual free space (instead of storing the total $\Theta(mn)$ size free space surface). Since the average Fréchet distance proved to be a good distance measure for trajectories, a global map-matching strategy based on this measure will be implemented. Further theoretical investigations will concentrate on how to compute the integral Fréchet distance and an average Fréchet distance with the normalization included within the optimization.

Acknowledgments

This research is supported in part by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-32645 DBGlobe project, the IXNILATHS project funded by the Greek General Secretariat of Research and Technology, and the Faculty Research Award Program at the University of Texas at San Antonio.

References

- [1] H. Alt, A. Efrat, G. Rote, and C. Wenk. Matching planar maps. *J. of Algorithms*, 49:262–283, 2003.
- [2] H. Alt and M. Godau. Computing the Fréchet distance between two polygonal curves. *Int. J. Comput. Geom. Appl.*, 5:75–91, 1995.
- [3] H. Alt and L. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation - a survey. In J.-R. Sack and J. Urrutia, editors, *Handbook of Computational Geometry*. Elsevier, 1999.
- [4] A. Arslan and O. Egecioglu. Efficient algorithms for normalized edit distance. *J. of Discrete Algorithms*, 1(1):3–20, 2000.
- [5] D. Bernstein and A. Kornhauser. An introduction to map matching for personal navigation assistants. Technical report, New Jersey TIDE Center Technical Report, 1996.
- [6] S. Brakatsoulas, D. Pfoser, and N. Tryfona. Practical data management techniques for vehicle tracking data. In *Proc. 21st ICDE conf.*, pages 324–325, 2005.
- [7] A. Civilis, C. S. Jensen, J. Nenortaite, and S. Pakalnis. Efficient tracking of moving objects with precision guarantees. In *Proc MobiQuitous conf.*, pages 164–173, 2004.
- [8] M. Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del circolo Matematico di Palermo*, 22:1–74, 1906.
- [9] J. Greenfeld. Matching GPS observations to locations on a digital map. In *Proc. 81th Annual Meeting of the Transportation Research Board*, Washington, DC, 2002.
- [10] R. Kuehne, R.-P. Schaefer, J. Mikat, K.-U. Thiessenhusen, U. Boettger, and S. Lorkowski. New approaches for traffic management in metropolitan areas. In *Proc. IFAC CTS Symposium*, 2003.
- [11] A. Leick. *GPS Satellite Surveying*. John Wiley & Sons, Inc., 1995.
- [12] G. S. Luecker. A data structure for orthogonal range queries. In *Proc. 19th Annu. IEEE Sympos. Found. Comp. Sci.*, pages 28–34, 1997.
- [13] A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *IEEE Trans. Patt. Anal. Mach. Int.*, 15(9):926–932, 1993.
- [14] D. Pfoser and C. S. Jensen. Capturing the uncertainty of moving-object representations. In *Proc. 6th SSD conf.*, pages 111–132, 1999.
- [15] M. Quddus, W. Ochieng, L. Zhao, and R. Noland. A general map matching algorithm for transport telematics applications. *GPS Solutions Journal*, 7(3):157–167, 2003.
- [16] D. Sankoff and J. Kruskal. *Time Warps, String Edits, and macromolecules: The Theory and Practice of Sequence Comparison*. Addison-Wesley, 1983.
- [17] F. van Diggelen. GPS accuracy: Lies, damn lies, and statistics. *GPS World*, 9(1):41–45, 1998.
- [18] H. Yin and O. Wolfson. A weight-based map matching method in moving objects databases. In *Proc. 16th SSDBM conf.*, pages 437–438, 2004.