A GRID-BASED SPATIAL INDEX FOR MATCHING BETWEEN MOVING VEHICLES AND ROAD NETWORK IN A REAL-TIME ENVIRONMENT

Meng Zhang[1], Masria Mustafa[1], Florian Schimandl[2] and Liqiu Meng[1]

[1] Department of Cartography,
Technische Universität München
Arcisstr. 21, 80333 Munich, Germany
{meng.zhang, masria.mustafa, meng@bv.tum.de

[2] Chair of Traffic Engineering and Control,
Technische Universität München,
Arcisstraße 21, 80333 Munich, Germany
florian.schimandl@vt.bv.tum.de

Abstract

Moving vehicles in a street network react to and reflect the actual traffic situation. Therefore, they can be considered as an important source of the dynamic traffic information, especially from large cities where it would be too expensive to seamlessly install stationary sensors for data acquisition. Without extra instrument to be set up along the roadway, traffic information can be derived from the behavior of moving vehicles at a low cost. Nevertheless, moving vehicles as such do not provide valuable information straightforwardly. Their scattered locations in a rather large space must be aligned with the street network, which is in essence a point-to-line matching task. So far, a large number of researchers worldwide have addressed various matching issues. Many of the reported results reveal a high matching rate with a high accuracy on certain data types or test areas. In case of matching individual vehicles with the streets along which they are dynamically moving, however, the matching quality is influenced not only by the accuracy, but also by the computing efficiency. It does not make any sense if the matching algorithm is unable to yield results before the vehicles move to substantially different locations. To realize the point-to-line matching in real time, the authors proposed an innovative grid-based spatial index for linear data on the basis of decomposition principles.

Keywords: spatial index, real-time environment, line segment, map matching

1 Introduction

Previous researches proved that it is an efficient and inexpensive way to acquire traffic information through map matching between the scattered positions of moving vehicles (equipped for the data collection of time-stamped location and speed) and the road network. This process can be implemented over large urban areas where the detectors are hardly possible to be fully activated. To date matching methodologies between moving vehicles and the road network were extensively discussed involving a number of algorithms of point-to-line map matching (Wu et al., 2007, Jong-Sun et al., 2001 and Quddus et al., 2003). Worthwhile to mention is that Lv et al. (2008) developed an approach to deduce matching errors caused by the GPS data, thus enhance the matching accuracy without having to raise the sampling rate of the moving vehicles. Neuland and Kürner (2007) applied a map-matching algorithm to automatically project the inaccurate GPS positions onto street vectors for the prediction of traffic states. Based on fuzzy logic, Quddus et al. (2006) provides a significant melioration over existing map matching algorithms to Ochieng at el. (2003), both in terms of identifying correct links and estimating the vehicle position on the links, especially in high density areas where the average distance between neighbouring roads is less than 100 metres. Furthermore, Yang et al. (2005) proposed a map matching algorithm which is geared for GPS data with relatively long polling time intervals (2~5 minutes).

Many hitherto reported matching algorithms revealed high matching rate with high accuracy on certain data types or sample sizes of selected test areas. Yet, the matching quality is influenced not only by the accuracy, but also by the computing efficiency. In the real-time applications, for example, it does not make any sense if the matching algorithm is unable to yield the matching results before the vehicles move to substantially different locations. Hence, a ‘good’ map matching algorithm should not only produce accurate results, but also do it fast. To enhance the matching efficiency, Marchal et al. (2004) demonstrated an interesting algorithm that merely relies on the GPS coordinates and the network topology. In this paper, however, we proposed a slightly different way to achieve the same intention of accelerating the computing speed. It uses a newly developed spatial index to organize the linear data (e.g. the road links). The proposed spatial index is simple, yet powerful in terms of enhancing the matching speed between points and lines.

2 Strategy

One of the key processes of matching between moving vehicles and the road network is to find all line segments within a given distance from a query point. It should be noted that the distance between a point and a line segment is different from that between a point and a whole line. The former can be expressed by equation [1]. In this equation, a line segment denoted as $l(p_i, p_j)$ (abbreviated as l_{ij}), where $p_i = (x_i, y_i)$ and $p_i = (x_j, y_j)$ are two end-points of l, $i \neq j$; thus, the distance between the query point $P_0 = (X_0, Y_0)$ and l_{ij} can be calculated as $Dis\ (P_0, l_{ij})$.

$$Dis\ (P_0, l_{ij}) = \begin{cases} \dfrac{\left|(y_j - y_i)\cdot X_0 + (x_i - x_j)\cdot Y_0 + (x_j \cdot y_i - x_i \cdot y_j)\right|}{\sqrt{(y_j - y_i)^2 + (x_i - x_j)^2}} & (if \quad \angle P_0 p_i p_j \le \dfrac{\pi}{2} and \angle P_0 p_j p_i \le \dfrac{\pi}{2}) \\ \sqrt{(x_i - X_0)^2 + (y_i - Y_0)^2} & (if \quad \angle P_0 p_i p_j > \dfrac{\pi}{2}) \\ \sqrt{(x_j - X_0)^2 + (y_j - Y_0)^2} & (if \quad \angle P_0 p_j p_i > \dfrac{\pi}{2}) \end{cases} \quad \ldots [1]$$

Usually, the street network is represented by vectors of coordinate pairs corresponding to their start and endpoints, which are ordered by their connectivity. Given a random query point in space, it is very inefficient to find its nearest line segment without using indexing structures. In order to accelerate the matching process, a grid-based spatial index for line segments was developed.

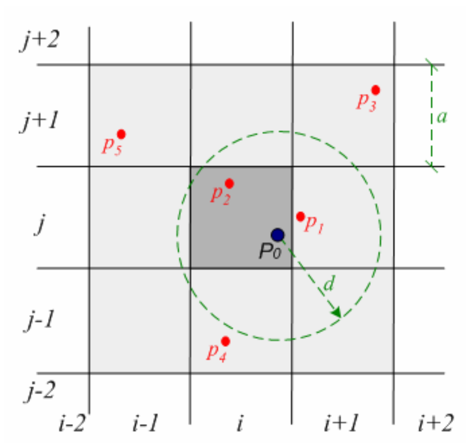
A spatial index is typically utilized in large spatial databases to optimize spatial queries, whereas the spatial index reported in this paper serves the purpose of accelerating the matching process between points and line segments. Two special constraints should be satisfied: (a) it should be easily established and implemented; and (b) it should lead to a maximum speed improvement of the matching process.

In the domain of spatial indexes, a grid is a regular tessellation of a manifold or 2-D surface divided into a series of contiguous cells, with each being assigned an identifier for indexing purposes. Various grid shapes have been proposed or are currently in use, such as square, rectangular, triangular, hexagon, diamond-shaped cells, and possibly many more. In practice, the construction of grid-based spatial indexes entails allocation of relevant objects to their position or positions in the grid, then creating an index of object identifiers versus grid cell identifiers for rapid access (Wikipedia 2009). In this paper, the grids based on *Square* are chosen to partition the large spatial space so that the linear data can be efficiently indexed.

2.1 Grid-based spatial indexes for point data

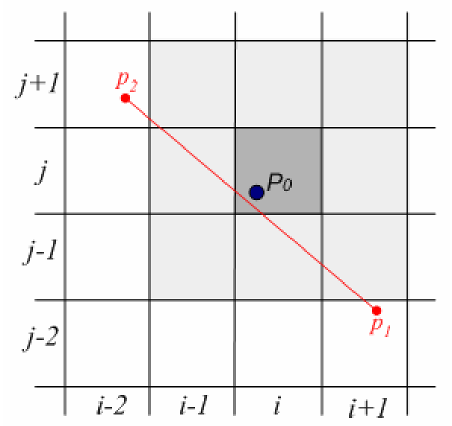
The grid-based spatial indexes for point data are the simplest methods to be used. They have been extensively tested for the organization of vector-based objects in large spatial databases. The understanding of the grid-based spatial indexes for point data will help us to create the spatial index for line segments.

As depicted in Figure 1, the whole spatial space is divided into a number of smaller square blocks. Each block can accommodate a varying number of points, while every point is inserted to one block that covers its entity. In the first step of finding all points within a given distance d to a query point P_0, the $block_{(i,j)}$ is confirmed as base region since it contains the query point. However, it may happen that the points which have a distance smaller than d to the query point P_0 are not necessarily located within the base region. For example in Figure 1, the point p_1 is very close to P_0 even though it falls outside *block* $_{(i,\,j)}$. Obviously, it is inadequate to restrict the search scope to the base region. To overcome this limitation, the *base region* and its neighbors $\{\, block_{I,J} \mid i-1 \le I \le i+1,\ j-1 \le J \le j+1,\ \ I \ne i, J \ne j, I, J \in N \,\}$ have to be treated together. This enlarged search scope is hereafter consistently referred to as *affected blocks*, viz. $\{\, block_{I,J} \mid i-1 \le I \le i+1,\ j-1 \le J \le j+1 \,\}$ in the subsequent illustrations. To ensure that the *affected blocks* cover all points within the tolerance distance *d* to the query point P_0 , the side of the square block has to be larger than the tolerance distance, i.e. $a > d$ in Figure 1.



Dark grey square: *base region*; shallow grey squares: *neighbouring blocks*

Figure 1. Grid-based spatial index for points data

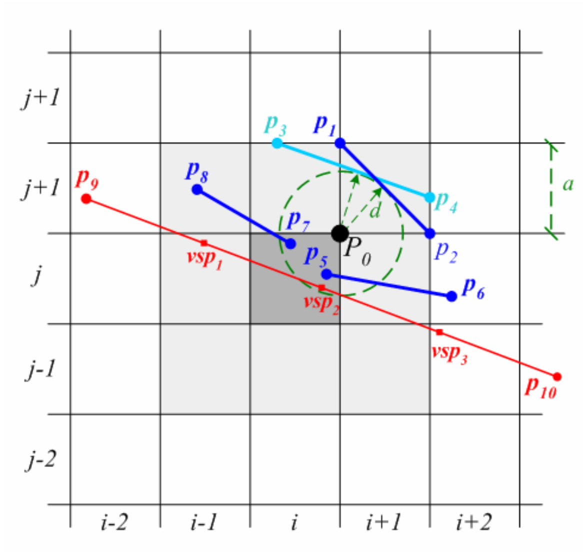


Dark grey square: *base region*; shallow grey squares: *neighbouring blocks*

Figure 2. Limitation of the grid-based spatial index for linear data

2.2 The index model for the organization of line segments

As mentioned earlier, the proposed spatial index in this paper is targeted to accelerate the process of finding all the line-segments $l_{i\,j}$ (viz. $l(p_i, p_j)$) within a distance ***d*** from a point $\boldsymbol{P_0}$, i.e. to identify a collection of the line segments $L(P_0) = \{\, l_{ij} \mid Dis(P_0, l_{ij}) \le d \,\}$ as fast as possible, where $Dis(P_0, l_{ij})$ represents the distance between point $\boldsymbol{P_0}$ and line segment $l_{i\,j}$ (ref. equation [1]). The grid-based spatial index described in section 2.1 can efficiently organize randomly distributed point objects into a hash directory. However, some artifacts may occur when this technique is directly implemented to deal with line segments. Applying the index techniques only to the start and end point of a line segment proves inadequate because the terminating points do not carry the actual shape information between them (Hoelf & Samet 1991). Often a line segment can stretch into a wide scope of space. If such an extensive line segment is indexed only to the blocks where its start and end points are located, its parts across the blocks cannot be handled. As demonstrated in Figure 2, the start point p_1 of the line segment l_{12} falls inside the lower-right *block* $_{(i+1,j-2)}$ and the end point p_2 belong to the *block* $_{(i-2,j+1)}$. Since neither *block* $_{(i+1,j-2)}$ nor *block* $_{(i-2,j+1)}$ is an *affected block* in $set(A\_block_{(i,j)}) = \{\, block_{I,J} \mid i-1 \le I \le i+1, j-1 \le J \le j+1 \,\}$, the line-segment l_{12} may be ignored during the searching process even though it is very close to the query point P_0. With regard to this kind of artifact, new methods are needed to index line segments.



Dark grey square: *base region*; shallow grey squares: *neighbouring blocks*
Figure 3. Grid-based spatial index for line segments

Figure 3 illustrates another example of a plane corresponding to a set of 25 square-grids in a 2-D Euclidean space. The query point is denoted here as P_0 and the side of the square grid is represented by *a* while the predetermined tolerance between the line segment and the query point is *d*. A worst case may occurs due to the following reasons:

(a) the query point P_0 is located on a corner of the *base region* (see the upper-right corner of the *block* $_{(i,j)}$ in Figure 3);
(b) the line-segment l_{12} is tangent to the circle with its centre located on query point P_0 and the radius is equal to tolerance *d*;
(c) the side of the square-grid *a* is equal to $\sqrt{2} \cdot d$;
(d) l_{12} is restricted by two corners of a *neighbouring block*, e.g. the upper-right *neighbour* of the base region *block* $_{(i,j)}$ (see *block* $_{(i+1,j+1)}$ in Figure 3); and
(e) the angle $\angle p_1 P_0 p_2$ is a right angle ($= \pi/2$).

This worst case can be more generally defined as follows: for a randomly given line segment *l*, if (1) it is not longer than l_{12}, i.e. $l \le l_{12} = 2d$, (2) it has a distance to the query point P_0 smaller than the tolerance *d, i.e.* $Dis\ (P_0, l) \le d$ (ref. equation [1]), and (3) the side of the square-grid is longer than $\sqrt{2} \cdot d$, i.e. $a > \sqrt{2} \cdot d$, then there must be at least one endpoint of *l* falling inside one of the *affected blocks*, see examples of l_{56} and l_{78} in Figure 3.

Indeed, a real-world network can possibly have line-segments longer than $l_{12} = 2d$. A practical solution to overcome this artifact is to introduce the following two decomposition principles:

i. During the partition process of the whole spatial space, the employed square grid should be larger than $\sqrt{2}d \times \sqrt{2}d$, where *d* means the tolerance distance between the query point *p* and line segment *l*;
ii. If the line-segment *l* is longer than *2d*, then a set of virtual specific points will be evenly inserted to *l*, e.g. the small square-shape points *vsp1, vsp2 and vsp3* along the line segment $l_{9,10}$ as illustrated in Figure 3. As the result, *l* can be split into several smaller proportions and each proportion has a length of no longer than *2d*. These virtual specific points, together with the start and end- point of the line segment *l*, will be treated as shape points for the further index calculations. Line segment *l* is assigned to all blocks when all shape points fall inside them.

Following these two decomposition principles, a spatial index model for the organization of line segments has been established. In this model, a *block* can accommodate a varying number of line segments; and vice versa a line-segment can also be occupied by different *blocks*. The data structure of the established spatial index is illustrated in Figure 4.

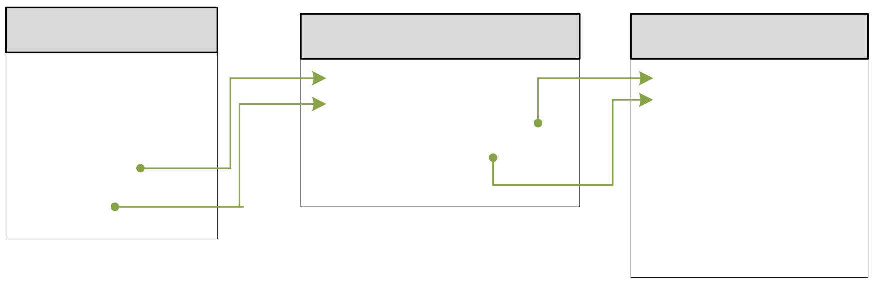


Figure 4. Data structure of the proposed spatial index for linear data

We examined the problem of identifying the line-segment collection of *L(P₀)* defined by $\{\ l_{ij} | Dis(P_0, l_{ij}) \le d \}$. The line segments are organized by the proposed spatial index established above. In the algorithm, the first step is to locate the base region containing the query point P_0. Then, all line segments occupied by the *affected blocks* are selected as the potential candidates to the collection *L*(P_0). In the example illustrated by Figure 3, besides the line segments l_{12}, l_{34}, l_{56} and l_{78}, $l_{9,10}$ will be also treated as one of the potential items in $L(P_0)=\{\, l_{ij} \,|\, Dis(P_0, l_{ij}) \leq d \,\}$ due to the fact that its virtual specific points (VSP) vsp_1 and vsp_2 are falling inside the *affected blocks*. Finally, each of the potential candidates is verified to see whether the distance to the query point P_0 is smaller than the pre-defined tolerance distance a or not. Thus, the speed of identifying the line segments within a tolerance distance from the query point has been dramatically enhanced. Instead of scanning every line segment distributed over the whole spatial space, only the potential candidates are calculated in the proposed model.

3 Experiment

In order to prove and demonstrate the effectiveness, the proposed spatial index model for line segments was tested in a number of on-going projects on the traffic state estimation by using moving vehicles as single source or by fusing the obtained data with other traffic data sources. As mentioned in Schimandl et al. (2009), these real-time applications consist of four key processes in general:

(a) receive dynamic vehicle positions over TCP/IP based on networks;

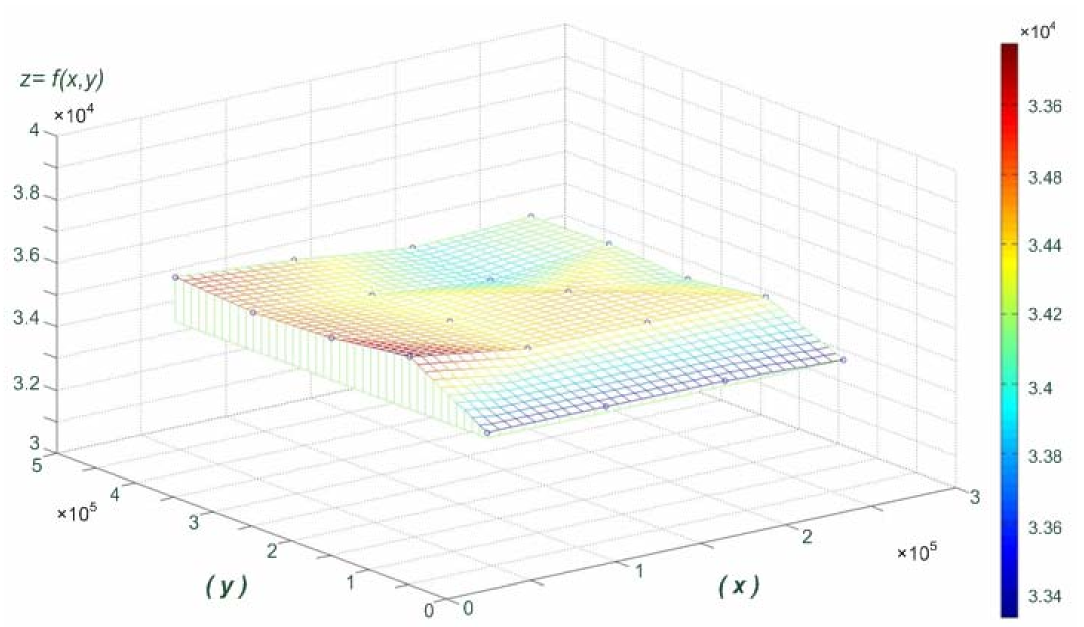
(b) find candidate road links;

(c) plausible matching; and

(d) determine or estimate the traffic state for each link.

Among them, the most time consuming processes is the projection of vehicle positions onto the correct link (line segment) of the road network, i.e. the efficiency of the “map matching” is crucial for the capability of processing the traffic data in real time. Using the proposed spatial index, the exponential computational complexity of the ‘map matching’ can be dramatically reduced. For instance, in order to find out the line segments (road links) within a tolerance distance to a moving vehicle position, all of the line-segments of the road network have to be calculated if the proposed spatial index is not used, that is $N_{L\text{-}Segment}$ cycles computations in total, where $N_{L\text{-}Segment}$ represent the number of the line segments. With help of the grid-based index, the computing intensity can be cut to around $9^* N_{L\text{-}Segment} / N_{Grid}$, where N_{Grid} is the total number of grids (viz. square blocks in Figure 3). The factor 9 represents the amount of the *affected blocks* (Xiong 2000). Taking a matching area with 10*10 km^2 (square kilometers) as an example, if the tolerance distance between the moving vehicle and the road links is equal to 35 m (meters), then the size of each grid can be set as 50 * 50 m^2 where the side of 50 m is a bit larger than $\sqrt{2} \times 35$ m. In this way, the whole matching region will be divided into (10000/50)^2 = 40000 square blocks and thus the searching time could become $40000/9 \approx 4.44 \times 10^3$ times faster theoretically.

The proposed spatial index has been tested within the urban road network of the city *Graz* (Austria) under real conditions, where the network data and the dynamical vehicle position data were easily accessible. Although we utilized a small urban road network in Graz which involves around 70000 line segments, the actual gain of implementing the proposed algorithm has been fully confirmed.



X-axis: size of the road network (number of line segments);   **Y-axis**: amount of moving vehicles;

Z-axis: matching speed (Vehicles/ second)

Figure 5. Computing efficiency on different matching experiments

Worthwhile to mention is that with the proposed spatial index, an increase of the network size will not significantly slow down the matching speed. Figure 5 demonstrates that the matching speed is more or less independent of the network size and the number of vehicle positions. In prove the hypothesis, we enlarged the road network of Graz to twice, triple and four times of its original size, disregarding the topologic information which is not needed for finding the candidate links. On each enlarged network graph, different experiments have been conducted with various numbers of the vehicle positions (100/200/300/400/500 thousands). As the matching speed keeps at around 34000 vehicles pro second, the road network with about 70000 or 280000 line segments requires very similar processing time to match 100000 vehicles. This is a very promising result because there is no significant difference in term of the processing speed no matter how large the network is and how many the vehicle positions are transmitted. Taking the advantage of our approach, the developed real-time system relying on the proposed spatial index has been successfully applied in some other cities, incl. two large cities in Germany and one mega city in China which covers an urban area of about 3700 km^2 and involves about 176000 road links with more than one million line segments. Due to the reason of 'industrial data protection', the names as well as the detailed information of these three cities are not allowed to be mentioned here.

4 Conclusions

In this paper a tricky grid-based spatial index for linear data based on the decomposition principles was proposed, which is a value-added approach to other standard point-to-line map matching algorithms. The spatial index proved to have considerably improved the computing speed for the matching between a point cluster and line segments, especially when dealing with large-scope road networks. The fast and accurate geometric alignment of a large number of vehicle positions to the individual streets of an extensive network provides an indispensable means for the real-time or nearly real-time estimation of the actual traffic state in urban areas. The proposed spatial index approach has been successfully applied in different systems for traffic state estimation in city Graz in Austria, two large cities in Germany and one mega city in China.

One limitation of the established spatial index is that the extremely large networks may overwhelm computer's physical memory if they are loaded at once as mentioned in (Xiong 2000). Currently, a matching area of about 170km^2 with 70000 link segments requires circa 10MB physical memory. If the size of the generated spatial index is larger than the available physically memory of the computer, some other strategies have to be applied. One possible way to solve this problem is to divide a large road-network, if it exceeds a predefined tolerance size, into several sub areas. Then, the point-to-line matching algorithm along with the proposed spatial index will be conducted in each sub area one by one, thus limit the physical memory intensity. Keeping in mind that the region division may cause unwanted border effects, we can set up a buffer surrounding each sub area, i.e. the actual searching scope is thus a little bit larger than each individual sub area (Zhang and Meng 2007). Future research will be conducted to explore this matter.

5 Reference

Hoelf, E.G., Samet, H. (1991): Efficient processing of spatial queries in line segment databases, Proc. of 2nd Symposium on Large Spatial Database (SSD' 91), Zurich, Switzerland, August 1991

Jong-Sun, P., Dong-Ho, S. & Tae-Kyung, S. (2001): Development of a map matching method using the multiple hypothesis technique, IEEE Intelligent Transportation Systems Conference. Oakland, CA, USA

Lv, W., Liao, W., Wu, D. & Xie, J. (2008): A New Road Network Model and Its Application in a Traffic Information System, Proc. of the Fourth International Conference on Autonomic and Autonomous Systems (ICAS 2008)

Marchal, F., Hackney, J. & Axhausen, K. W. (2004): Efficient Map-Matching of Large GPS Data Sets - Tests on a Speed Monitoring Experiment in Zurich, volume 244 of Arbeitsbericht Verkehrs und Raumplanung

Neuland, M. & KÜRNER, T. (2007): Analysis of the impact of map-matching on the accuracy of propagation models, Advances in Radio Science, Volume 5, pp.367-372

Ochieng, W.Y., Quddus, M.A. & Noland, R.B. (2003): Map-Matching in Complex Urban Road Networks, Brazilian Journal of Cartography (Revista Brasileira de Cartografia), 55 (2), 1-18

Quddus, M. A., Ochieng, W. Y., Zhao, L. & Noland, R. B. (2003): A general map matching algorithm for transport telematics applications, GPS Solutions, Volume 7, pp. 157-167

Quddus, M. A., Noland, R. B., & Ochieng, W. Y. (2006): A High Accuracy Fuzzy Logic Based Map Matching Algorithm for Road Transport, Journal of Intelligent Transportation Systems: Technology, Planning, and Operations, 10 (3), pp.103-115

Schimandl, F., Zhang, M., Mustafa, M., Meng, L. (2009): Real time application for traffic state estimation based on large sets of floating car data, International Scientific Conference on Mobility and Transport - ITS for larger Cities, May 12~13th, 2009, Munich, Germany

Wikipedia (2009): http://en.wikipedia.org/wiki/Grid_(spatial_index), accessed on 2009-07-02

Wu, D., Zhu, T., Lv, W. & Gao, X. (2007): A Heuristic Map-Matching Algorithm by Using Vector-Based Recognition, International Multi-Conference on Computing in the Global Information Technology, Guadeloupe, French Caribbean

Xiong, D. (2000): A three-stage computational approach to network matching, Transportation Research Part C: Emerging Technologies Volume 8, pp. 71-89

Yang, J.-S., Kang, S.-P. & Chon, K.-S. (2005): The Map Matching Algorithm Of Gps Data With Relatively Long Polling Time Intervals, Journal Of The Eastern Asia Society For Transportation Studies, Volume 6, Pp. 2561 - 2573

Zhang M. & Meng L. (2007): An iterative road-matching approach for the integration of postal data, Computers, Environment and Urban Systems, volume 31/5 pp. 598 - 616