Transportation Research Part C 66 (2016) 176–194

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Estimating the most likely space–time paths, dwell times and path uncertainties from vehicle trajectory data: A time geographic method

Jinjin Tang [a], Ying Song [d], Harvey J. Miller [b], Xuesong Zhou [c,*]

[a] School of Traffic and Transportation, Beijing Jiaotong University, Beijing 100044, China
[b] Department of Geography, The Ohio State University, Columbus, OH 43210, USA
[c] School of Sustainable Engineering and the Built Environment, Arizona State University, Tempe, AZ 85287, USA
[d] Department of Geography, Environment and Society, University of Minnesota, MN 55455, USA

ARTICLE INFO

Article history:
Received 27 January 2015
Received in revised form 25 July 2015
Accepted 25 August 2015
Available online 9 September 2015

Keywords:
GPS map matching
Traffic state estimation
Dynamic shortest path
Uncertainty estimation

ABSTRACT

Global Positioning System and other location-based services record vehicles' spatial locations at discrete time stamps. Considering these recorded locations in space with given specific time stamps, this paper proposes a novel time-dependent graph model to estimate their likely space–time paths and their uncertainties within a transportation network. The proposed model adopts theories in time geography and produces the feasible network–time paths, the expected link travel times and dwell times at possible intermediate stops. A dynamic programming algorithm implements the model for both offline and real-time applications. To estimate the uncertainty, this paper also develops a method based on the potential path area for all feasible network–time paths. This paper uses a set of real-world trajectory data to illustrate the proposed model, prove the accuracy of estimated results and demonstrate the computational efficiency of the estimation algorithm.

1. Introduction

Emerging mobile computing and sensor techniques have improved capabilities to collect and process real-time traffic data for traffic state monitoring and management applications. For example, Global Positioning System (GPS)-based in car navigation has reached a significant level of penetration rate, and most smart phones are equipped with GPS receivers with high-speed data communication links. Nevertheless, current vehicle location data are still associated with location errors, typically within a wide range of 5–300 m given the ground-truth vehicle trajectory. A critical data processing component in emerging Big Data applications is how to systematically use latitude, longitude, and time stamps of a single probe vehicle or a set of probe trajectories to estimate traffic states at different scales.

In this paper, we present a time-geography based approach (Hägerstrand, 1970) to consider not only the geometry and topology of the road network, but also the time attributes in available GPS samples. We also incorporate a space–time network-based representation adapted from the Time-Dependent Shortest Path problem (TDSP). Essentially, a sequence of GPS traces with both location and timestamp information can be mapped as a space–time trajectory or path. Within a space–time network, there are a large number of possible paths with different degrees of spatial and temporal distance to the vehicle trajectory records. Our approach aims to find the most likely network–time path that minimizes the total

* Corresponding author. Tel.: +1 480 9655827.
E-mail addresses: jinjintang@hotmail.com (J. Tang), yingsong@umn.edu (Y. Song), miller.81@osu.edu (H.J. Miller), xzhou74@asu.edu (X. Zhou).

http://dx.doi.org/10.1016/j.trc.2015.08.014
0968-090X/

map-matching distance among all possible alternatives. Our method also allows estimation of dwell and detour times at intermediate nodes and the uncertainties associated with the likely paths. The proposed mathematical programming model can integrate with various sensor data sources and finds the optimal solution that takes into account the distance measure at different time stamps of driving traces.

This paper is organized as follows. Section 2 provides the background to the trajectory map-matching problem, alternative solutions and the features of our time geographic approach. Section 3 illustrates the proposed space time network representation for finding the most likely network–time paths, followed by an introduction on the time dependent shortest path model in Section 4. Section 5 uses illustrative example to further explain the fundamental of the model. Section 6 presents dynamic programming algorithms to find the time dependent shortest path with generalized cost functions specific to the network–time path estimation problem. After a discussion on uncertainty quantification for potential accessible space–time nodes within the estimated paths in Section 6.3, numerical experiments on a simple network and a real-world network are presented in Section 7. Section 8 concludes the paper with summary comments and future research steps.

2. Background

Many traffic management and planning agencies have started using the vehicle trajectory data for estimating travel time, route choice behavior and activity location patterns. The traffic states of interest include the most likely paths and related traffic speed and density on the transportation network. In this research, we are interested in providing a time-expanded graph modeling framework for fully utilizing location-based data and sensor measurements from different sources, such as automated vehicle counts, virtual detection lines as well as fixed detectors. Within the last 30 years, a large number of algorithms have been proposed to the general problem of GPS map matching, which aims to find the closest or most likely matched link sequence and travel speed. These methods can be broadly categorized in the following.

(i) *Geometric map-matching algorithm*. Geometric map-matching algorithms compare raw GPS points with the geometries of the underlying road network, in order to obtain a sequence of likely links (Greenfeld, 2002). A simple approach along this line is to match each point with the nearest road node (Bernstein and Kornhauser, 1998), while other sophisticated methods involve point-to-curve or curve-to-curve geometric distances (White et al., 2000). As this approach generally does not consider connectivity, it is possible that the matched links are disconnected from one other. To address this issue, Fu et al. (2004) proposed a hybrid map-matching algorithm by examining the geometry of the road network and fuzzy comprehensive judgment. Kong et al. (2013) recently integrated curve-fitting-based method and a vehicle-tracking-based method to estimate traffic states from GPS probe data along a path without detour.

(ii) *Topological map-matching algorithm*. An approach proposed by Greenfeld (2002) aims to find a topologically feasible (but time-invariant) path through the road network, with the arc weights in the related topological path-search algorithm without considering any heading or speed information from GPS data. Meng (2006) further considered other topological features such as road intersections, road curvature and road connections. Some other topological map-matching algorithms (Yin and Wolfson, 2004; Yang et al., 2003) utilize the connectivity and contiguity information of road networks to improve link identification rates in GPS map matching applications.

(iii) *Statistical algorithms*. Honey et al. (1989) first introduced a probability-based algorithm to clearly definite an elliptical or rectangular confidence region around a position. Zhao (1997) suggested that the error region can be derived from the error variances associated with GPS positions. To further quantify and determine map-matching probabilities given noisy data, this type of algorithms have integrated various statistic methods, to name a few, Kalman Filters and Extended Kalman Filters (e.g. Kim et al., 2000; Krakiwsky et al., 1988; Obradovic et al., 2006;Jo et al., 2012), fuzzy logic (e.g. Quddus, 2006; Zhao, 1997; Syed and Cannon, 2004), Bayesian inference (e.g. Pyo et al., 2001), and Particle Filter (Peker et al., 2011). Within an optimal filtering framework, the above algorithms recursively estimate the likely path and error covariance matrix associated with the estimated states under different measurement error assumptions.

Beside these time-invariant algorithms, studies have started to account for travel time beside distance of vehicle trajectories. Aiming to assist informed traffic management decisions, traffic state estimation techniques are often used to estimate end-to-end trip travel time and congestion levels of the traffic system using heterogeneous data sources. Early studies such as Szeto and Gazis (1972), Cremer and Papageorgiou (1981), and recent studies such as Wang and Papageorgiou (2005), Muñoz et al. (2003), Sun et al. (2003) and Work et al. (2010) focus on how to use detector data and various traffic state filtering methods to estimate traffic flow, density and queue lengths on each link segment of the freeway corridor. Herrera and Bayen (2010) also proposed a novel method of virtual trip lines to estimate traffic states based on trajectory data from both arterial and freeway corridors. A recent study by Deng et al. (2013) extended Newell's three-detector method as a stochastic measurement equation to quantify the value of Automated Vehicle Identification (AVI), Automated Vehicle Location (AVL) and point sensor measurements.

While significant progress has been made in recent years, there are still a number of challenging questions to be addressed to systematically estimate vehicle paths within a transportation network based on trajectory data. First, many GPS map-matching algorithms mainly focus on spatial attributes of GPS records within a short time window, in conjunction

with the geometry and topology information of the road network. A desirable traffic state estimation model in this context should utilize both space and time attributes from AVI, and AVL/GPS sample data. Many new applications are looking for a much richer set of traffic states beyond the likely link sequence, such as the durations of dwelling or stationary activities, link traveling time and waiting time at intersections. The above information is critical not only for many traffic engineering projects (such as optimizing traffic signal timing) but also for a wide range of accessibility based multi-modal transportation planning decisions (e.g., evaluating impact of parking policies on activity duration at downtown areas).

Second, many existing GPS map matching algorithms are very suitable for high-frequency location data (say every 1 s). It is very challenging yet valuable to construct a well-defined optimization or estimation framework that can be fully utilize location-based data sources with different sampling time and spatial resolutions. For example, the sampling interval for cell phone tower data could be significantly large, with a range of 30 s to 5 min (Thiagarajan et al., 2011). In this case, if one only simply considers the distance from GPS points to a single node or a single link, then it could fail to capture several major factors in real-world trajectories, such as the link-to-link connectivity, possibility of dwelling and stationary activities.

Third, a desirable traffic state estimation method should not only identify the most possible map-matched links but also describe the uncertainty of the estimation results, in a similar way to the mean estimate and the corresponding estimation error variance–covariance matrix in a general Kalman filtering framework. In the case of path matching, we are interested in several key questions. (1) If there are alternative likely paths available within a similar estimation error range; (2) How many of those paths can be found, and (3) Is it possible to increase the sampling interval or add additional location sensors to reduce the estimation uncertainty?

With a unified time-expanded graph modeling framework across different types of location-based vehicle measurements, the methods in this paper can integrate heterogeneous data sources to improve estimation quality and address specific needs from high-quality traffic data mining applications. Our goal is to develop a theoretically sound and computationally efficient method for both offline and real-time applications. Our proposed method can further calculate the uncertainty boundaries of estimated paths in a transportation network. With a solid uncertainty estimation results, one can evaluate the quality impact of GPS sampling intervals, market penetration rates of probe vehicles, as well as the spatial resolution of underlying networks.

3. Conceptual foundation, problem formulation and illustrative example

Time geography represents individual's actual and potential mobility using *space–time paths* and *space–time prisms* respectively (Hägerstrand, 1970). The sample space–time path in Fig. 1(a) illustrates a traveler's movements among activity locations with respect to time. The slope of the path segment indicates the moving speed: a steeper line means more time is spent between two locations and therefore lower speed; a vertical line corresponds to conducting stationary activity at the same location through time; and the maximum achievable speed is determined by the individual's mobility level. A space–time prism is the envelope of all feasible space–time paths between two activity locations given the time budget for traveling and the maximum achievable speed (Fig. 1b). Its projection the space constitutes *the potential path area* (PPA) that delimits all feasible routes in space.

In the real world, individuals' movements are often confined to spatial networks such as those corresponding to transportation systems and major activity locations. Accordingly, one can define *network–time paths* and *network–time prisms*

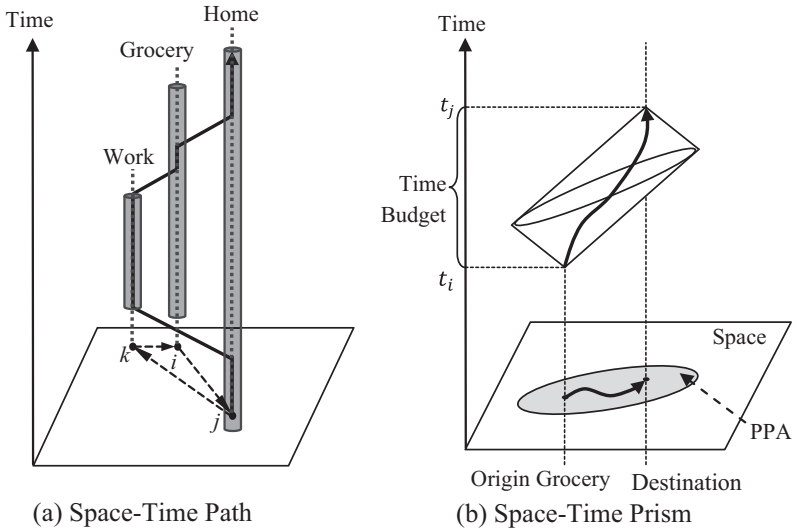


Fig. 1. Illustration of space–time path and space–time prism.

to take into account additional constraints such as geometry, connectivity, and speed limits imposed by spatial networks (Miller, 1991; Kuijpers and Othman, 2009). Each location within the network has a forward optimal label for the least time cost from the origin to that location, and a backward optimal label for the least time cost from that location to the destination. A location within the network is in the PPA if and only if its forward and backward optimal label costs together do not exceed the time budget for traveling. The network time prism has been applied to study individual's potential mobility (e.g. Downs and Horner, 2012) and create people-based accessibility measures (e.g. Kwan, 1998; O'Sullivan et al., 2000).

In this paper, the specific traffic state estimation problem aims to estimate most likely network–time paths, given a sequence of vehicle location records available from, AVL and AVI sensors. The underlying transportation network (serving as spatial constraints) typically contains a set of geographically referenced nodes (such as freeway merge/diverge nodes and arterial intersections), and a set of directed links in different road type categories, for example, freeway, highway and arterial streets. Without the loss of generality, we assume the geometric curvature information of a road has been coded through a vector of node coordinates. If an original road link is represented as a sequence of curved road segments, then we will accordingly decompose the link to a sequence of straight-line links in our model.

The constructed time-expanded network can be defined as $G = (V, A, T)$, where V is the set of time-dependent vertexes, of cardinality $n \times T$, and A is the set of time-dependent arcs corresponding to any feasible transition between vertexes. The discretized time-expanded network permits us to draw upon the many algorithms of time-dependent shortest path and other related dynamic network flow algorithms that can exploit the special features of the network-path map-matching problem. The objective of the estimation problem is to find the most likely path(s) in the time-dependent network that can minimize the total traffic state estimation error while subject to network connectivity and feasible travel time requirements. Tables 1 and 2 give the related notation, input parameters and estimation variables of the corresponding problem.

Let us first consider a hypothetical 3-node network and 6 raw location records of a vehicle, shown in Fig. 2(a). Nodes and records detailed coordinate information is given in Tables 3 and 4, respectively. Obviously, there are two potential paths for a vehicle traveling from node i to node k, namely P1: $i \to j \to k$, and P2: $i \to k$.

In our notation system, a vertex is always associated with a time index and is denoted as $i(t)$ for node i at time t. Accordingly, an arc is also indexed by time interval numbers and is expressed in terms of four indices (i, t, j, t'), and arc cost $c_{ij}^{t,t'}$ is associated with a traversal action leaving from node i to node j, with an entering time t and an exit time t'.

The constructed time-expanded network has a time dimension of $T = 6$ for each node, e.g., the set of vertex is $V = \{i(1), \ldots, i(t), \ldots, i(6), j(1), \ldots, j(6), k(1), \ldots, k(6)\}$.Using node i as an example, the arc set A has two categories of arcs can be originated from a vertex: (i) traveling arcs that move straight to the next node ($j \neq i$); (ii) dwelling arcs that stop at the same node till next time stamp $t + 1$. As shown in Fig. 3a, arc $(i, 1, i, 2)$ is an example of a dwelling arc, while the other arcs, e.g. arcs $(i, 1, j, 6)$ and arc $(i, 1, k, 2)$, are traveling arcs.

Table 5 further illustrates the possible network–time paths (NTP) in the sample network. Along the time period from second 1 to second 6. Fig. 4 aims to display network–time paths 1 and 2 in three-dimensional geographical space. In addition, paths 1 and 2 have the same sequence of nodes (i, j, k), but different dwelling times at different locations. NTP δ has a different node sequence from i to j.

A typical GPS map matching method calculates the shortest distance from a point to a line using the orthogonal distance function that finds the length of the straight line segment that intersects that line at a right angle. In contrast, our proposed method calculates the distance between two geometric locations referenced at the same time stamp to fully consider the temporal dimension of available information.

4. Time-dependent least cost path model for joint estimation of travel time and link sequence

The key question in the previous example is how to minimize the overall estimation cost for possible time-dependent paths. This section constructs an optimization model for the generic network–time path estimation problem as a

Table 1
Notations and input parameters.

Symbol	Definition
i, j, k, l	The indexed nodes, $i, j, k, l \in N$, N is the set of road network nodes, the cardinality of N is n
(i, j)	The indexed traffic link between adjacent nodes i and j, cardinality is m
t, t', t'', τ	The indices of modeling time stamps, $t = 1, 2, \ldots, T$, T is the number of time stamps in the expanded network for estimation, $t' > t$, $t'' > t'$
x_i, y_i	The latitude and longitude of node i
$i(t), j(t), k(t)$	The vertex indices in space–time network, for nodes i, j and k at time t, with respectively
(i, t, j, t')	The arc index in space–time network, leaving from node i to node j, with an entering time t and an exit time t'
$BS(j, t')$	Set of incoming arcs going to vertex $j(t')$, i.e., backward arc set including a number of vertexes $i(t)$
$FS(j, t')$	Set of outgoing arcs coming from vertex $j(t')$, i.e., forward arc set including a number of vertexes $k(t'')$
g_x^τ, g_y^τ	The recorded vehicle longitude and latitude at time τ
$d(i, t)$	The Euclidean distance from the recorded (g_x^t, g_y^t) to the node i
$d_{ij}^{t,t'}(g_x^\tau, g_y^\tau)$	The Euclidean distance from the recorded space–time location (g_x^τ, g_y^τ) to the space–time arc (i, t, j, t')
$c_{ij}^{t,t'}$	The cost of arc (i, t, j, t') from upstream node $i(t)$ to downstream node $j(t')$
FFTT(i, j)	Free flow travel time of link (i, j)
MaxTT(i, j)	The maximum travel time of link (i, j), MaxTT <= T

Table 2
Estimation variables.

Symbol	Definition
$z_{i,j}^{t,t'}$	0–1 binary variables, = 1, if a vehicle passes link (i,j) from time stamp t at the upstream node i to t' at downstream node j, = 0, otherwise

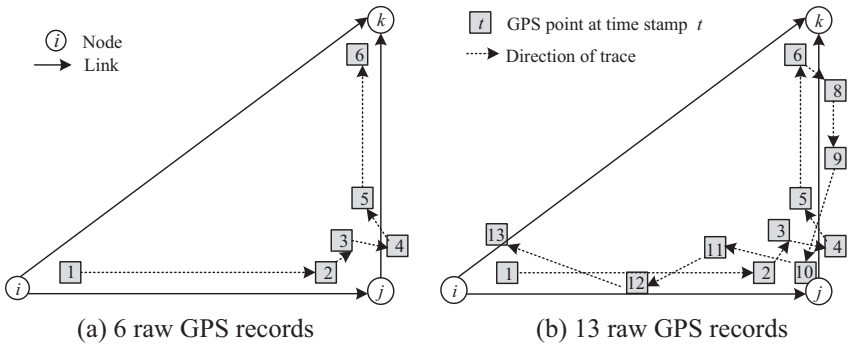


Fig. 2. Hypothetical 3-node network based on two-dimensional network.

Table 3
Node coordinates of road network.

Node ID	x	y
i	1	1
j	5	1
k	5	5

Table 4
Locations records of vehicle at different timestamps.

Timestamp	g_x^t	g_y^t
1	1.1	1.05
2	4.8	1.05
3	5.1	1.1
4	4.8	1.08
5	4.8	2.3
6	4.95	4.9

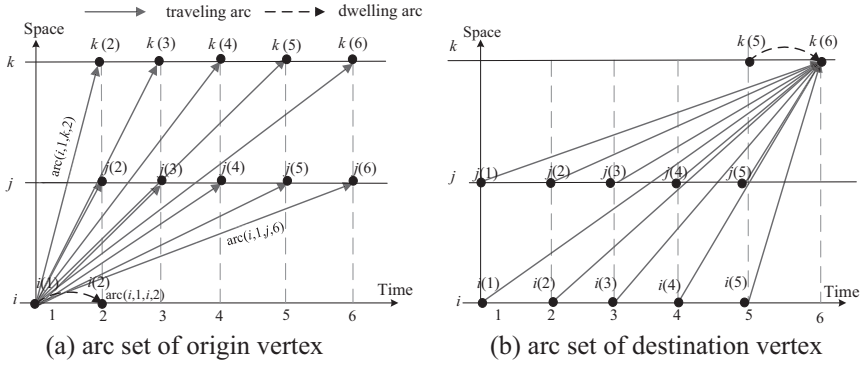


Fig. 3. Time-expanded network presentations along different arcs at and origin and destination vertexes.

time-dependent least-cost path problem. This optimization framework assumes the arc cost has been calculated for each feasible arc in the time-expanded network, and the optimal path finding algorithm to be presented in the next subsection is able to find a network–time path solution that minimizes the total traffic state estimation errors among all possible solutions.

Table 5
Possible network–time paths.

Time index (second)	1	2	3	4	5	6
NTP 1 (most likely network–time paths)	i	j	j	j	→	k
NTP 2	i	→	j	j	→	k
NTP 3	i	j	j	j	j	k
…	…	…	…	…	…	…	…
NTP δ	i	→	→	→	→	k

→ represents matched coordinate located in the middle position between upstream node and downstream node.

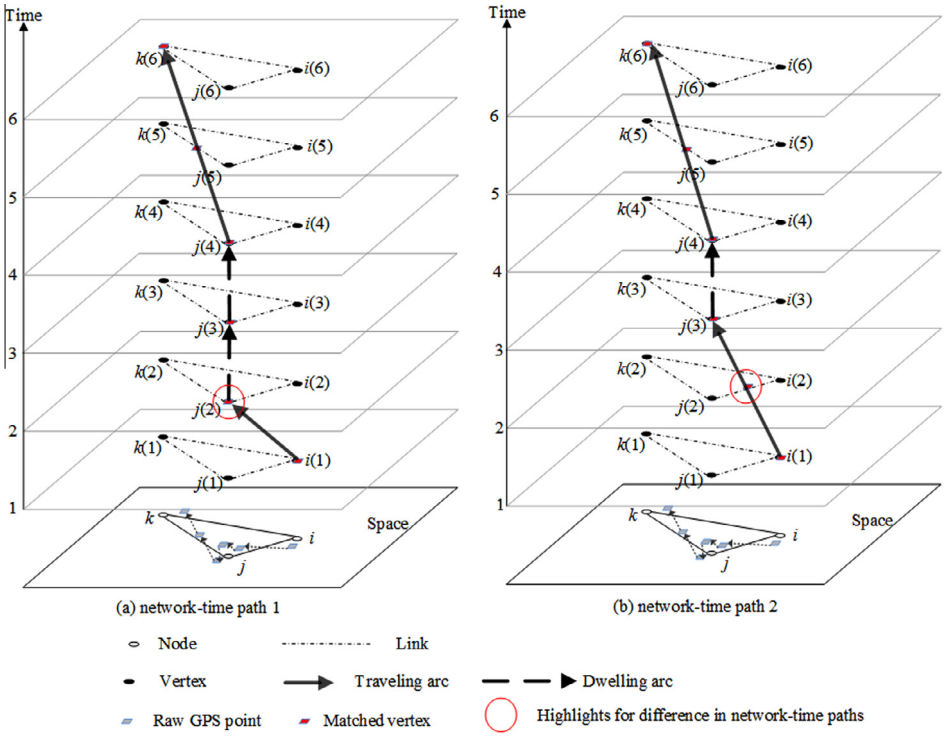


Fig. 4. Different network–time path solutions and calculating node-specific matching distance for 6 GPS points on a 3-node network based on three-dimensional network (projection of space–time path on the space plane).

4.1. Data sources and optimization model

The optimization approach in this paper can accommodate two typical data sources for vehicle movement:

(i) AVL data, containing semi-continuous location points $\{g_x^t, g_y^t\}$ for $t = 1, \ldots, T$, for example, from GPS navigation devices or cell phone tower location data, with different degrees of measurement errors.
(ii) AVI data, from a pair of vehicle tag readers (e.g., Bluetooth readers) at time indices t and t' at nodes i and j, corresponding to $\{g_x^t, g_y^t\}$, $\{g_x^{t'}, g_y^{t'}\}$, respectively.

We first define a binary variable $z_{i,j}^{t,t'}$ to indicate if a vehicle passes link (i,j) from time stamp t at the upstream node i to t' at downstream node j. With the constructed time-dependent network, we can easily establish the following integer programming model with objective function (1) and flow balance constraints (2–4) at different types of vertices, namely trajectory origin, trajectory destination and the other time-indexed nodes.

The objective function minimizes the total estimation error between observed data points and estimated nodes with the corresponding time indices, that is, $\sum_{i\in \text{Path}} \sum_{\tau=1}^{T} d(i,\tau)$, where $d(i,\tau)$ is the time-index distance measure between observed location to the mapping node at sampling time index τ. Depending on the estimation error distribution (e.g. normally distributed vs. uniformly distributed), the error measure could be squared or absolute distance between a pair of locations. As the space–time network is constructed using modeling time index t, we need to link these two time index systems together.

If the two systems are consistent with the same resolution, one can simply consider $\sum_{i\in Path}\sum_{t=1}^{T} d(i,t)$ as the objective function. Interested readers are referred to a book by Ahuja et al. (1993) for a systematic discussion on flow conservation constraints within a network flow programming framework, and a recent paper by Yang and Zhou (2014) on how to formulate multidimensional space–time network models.

Without loss of generality, we first consider there are high frequency of location data available across all time t, so we can consider the following equivalent objective function with respect to each space–time arc.

$$Min\ C = \sum_{(i,j)\in A}\sum_{t=1}^{T}\sum_{t'>t}^{T}\left(c_{i,j}^{t,t'} \times z_{i,j}^{t,t'}\right) \tag{1}$$

Subject to:

(Flow balance constraints at origin node r and time $t = 1$, namely vertex $r(1)$ as shown in Fig. 3a.)

$$\sum_{j,t'>t} z_{r,j}^{t,t'} = 1 \quad t = 1 \tag{2}$$

(Flow balance constraints at the destination node s and time $t' = T$, namely vertex $s(T)$ as shown in Fig. 3b.)

$$\sum_{i,t<t'} z_{i,s}^{t,t'} = 1 \quad t' = T \tag{3}$$

(Flow balance constraints at intermediate vertex $j(t')$):

$$\sum_{i(t)\in BS(j,t'),t<t'} z_{i,j}^{t,t'} - \sum_{k(t'')\in FS(j,t'),t'<t''} z_{i,j}^{t',t''} + z_{j,j}^{t'-1,t'} - z_{j,j}^{t',t'+1} = 0 \quad \forall j,\ \forall t' \tag{4}$$

Constraints (2)–(4) ensure flow balance on the network at the origin *vertex* $i(1)$, destination *vertex* $s(T)$ and intermediate *vertex* $j(t')$, respectively. It should be remarked that, $BS(j,t')$ is the set of incoming arcs going to vertex $j(t')$, and $FS(j,t')$ is the set of outgoing arcs coming from vertex $j(t')$. As an example, Fig. 5 shows a backward vertex set $BS((j,3)) = \{(i,1), (i,2), (j,2)\}$ and a forward vertex set $FS((j,3)) = \{(k,4), (k,5), (k,6), (j,4)\}$.

4.2. Defining arc cost for traffic measurement in a space–time network

The formulation challenge within an expanded network structure is how to properly define and calculate arc costs for the network–time path estimation problem. To estimate the cost of each arc (i,t,j,t') along the link (i,j), we have the following generalized time-indexed location-to-link distance equation:

$$c_{i,j}^{t,t'} = \sum_{\tau=t}^{t'} d(i,j,\tau) \tag{5}$$

where

$$d(i,j,\tau) = \begin{cases} d(i,t) & \tau = t \\ d(j,t') & \tau = t' \\ \bar{d}_{i,j}^{t,t'}\left(g_x^{\tau}, g_y^{\tau}\right) & otherwise \end{cases} \tag{6}$$

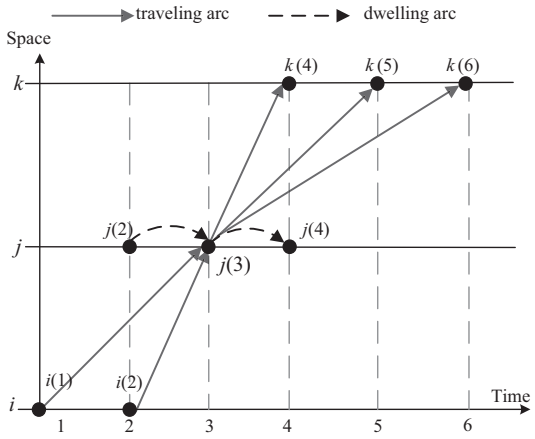


Fig. 5. Time-expanded network presentations along different arcs at one vertex.

with the time-indexed location-to-node distance measure function:

$$d(i,t) = \sqrt{\left(x_i - g_x^t\right)^2 + \left(y_i - g_y^t\right)^2} \tag{7}$$

and the approximate distance from GPS positions to intermediate points of a line (network arc):

$$\bar{d}_{i,j}^{t,t'}\left(g_x^\tau, g_y^\tau\right) = \sqrt{\left[\left(\frac{\tau - t}{t' - t} \times (x_j - x_i) + x_i\right) - g_x^\tau\right]^2 + \left[\left(\frac{\tau - t}{t' - t} \times (y_j - y_i) + y_i\right) - g_y^\tau\right]^2} \tag{8}$$

As a special case, if we do not consider the distance from GPS positions, we can have a simplified time-indexed location-to-link distance measure equation:

$$c_{i,j}^{t,t'} = d(i,t) + d(j,t') \tag{9}$$

Using Eq. (8), we compute the straight-line distance $d(i,j,\tau)$ from the map-matched coordinate to GPS point $\left(g_x^\tau, g_y^\tau\right)$ at time τ. The detailed calculation steps will be illustrated in Fig. 6 in the following section. Furthermore, Eq. (5) can be applied to a special case for waiting arc $(i,t,i,t{+}1)$ (for non-traveling activity locations or stopping before signal lights) as shown in Eq. (10):

$$c_{i,i}^{t,t+1} = d(i,t) + d(i,t+1) \tag{10}$$

For simplicity, Eq. (5) will double-count the location-to-node distances (with respective to intermediate node j in consecutive links (i,j) and (j,k)). Because different paths would have the same degree of distance "double-counting", the proposed optimization objective function does not lead to biased results.

5. Illustrative example

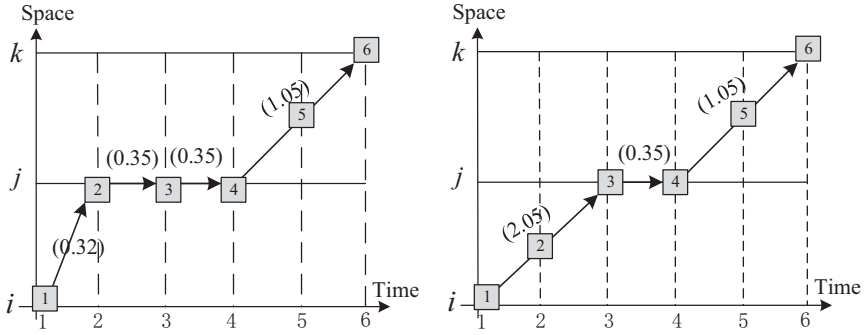
We now use Figs. 4 and 6 to illustrate how different cost values are calculated in a hypothetical network. As shown in Figs. 4(a) and 6(a), a space–time path (TP1) in the time-expanded network contains 5 vertices, including vertex $i(1)$, vertex $j(2)$, vertex $j(3)$, vertex $j(4)$ and vertex $k(6)$, and 4 arcs including arc$(i,1,j,2)$, arc$(j,2,j,3)$, arc$(j,3,j,4)$ and arc$(j,4,k,6)$. This solution reads as a sequence of events: the vehicle arrives at node i at time 1, arrives at node j at time 2, leaves node j at time 4, and arrives at node k at time 6. The space–time path shows that a vehicle pass the path (P1: $i-j-k$). This implies matching record 1 to node i, matching records 2, 3, and 4 to the same node j, matching record 5 to an intermediate point on link (j,k), and matching record 6 to node k.

As GPS points 1, 2, 3, 4 and 6 have been matched directly to nodes, we use Eq. (7) to calculate the vertex distance:

$$d(i,t=1) = \sqrt{(1-1.1)^2 + (1-1.05)^2} \approx 0.11.$$

$$d(j,t=2) = \sqrt{(5-4.8)^2 + (1-1.05)^2} \approx 0.21.$$

$$d(j,t=3) = \sqrt{(5-5.1)^2 + (1-1.1)^2} \approx 0.14.$$



(a) First path solution with cost = 2.07 Legend: (arc cost)

(b) Second path solution with cost = 3.45

Fig. 6. Illustrations of different cost values calculated for different paths.

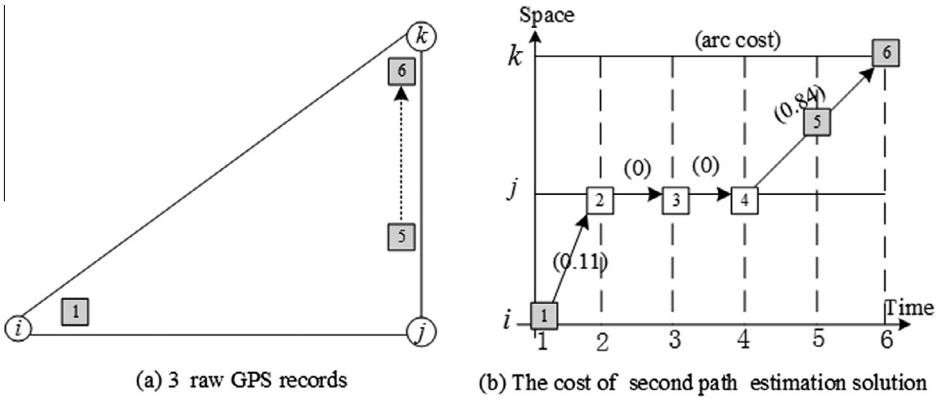


Fig. 7. The cost values are calculated in the hypothetical network under no location data at time stamps 2, 3 and 4.

$$d(j,t=4)=\sqrt{(5-4.8)^2+(1-1.08)^2}\approx 0.21.$$

$$d(k,t=6)=\sqrt{(5-4.95)^2+(5-4.9)^2}\approx 0.11.$$

As a special case, record 5 is matched at an intermediate location within link (j,k), so we need to calculate those vertex cost the vertex cost $\bar{d}^{4,6}_{j,k}\left(g^5_x,g^5_y\right)$ as Eq. (8) leads to

$$\bar{d}^{4,6}_{j,k}\left(g^5_x,g^5_y\right)=\sqrt{\left[\frac{5-4}{6-4}\times(x_k-x_j)+x_j-4.8\right]^2+\left[\frac{5-4}{6-4}\times(y_k-y_j)+y_j-2.3\right]^2}\approx 0.73$$

Finally, as plotted in Fig. 4b, all arcs cost can be computed as:

$$c^{1,2}_{i,j}=d(i,1)+d(j,2)\approx 0.11+0.21=0.32,$$

$$c^{2,3}_{j,j}=d(j,2)+d(j,t=3)\approx 0.21+0.14=0.35,$$

$$c^{3,4}_{j,j}=d(j,t=3)+d(j,t=4)\approx 0.14+0.21=0.35,$$

$$c^{4,6}_{j,k}=d(j,t=4)+\bar{d}^{4,6}_{j,k}\left(g^5_x,g^5_y\right)+d(k,t=6)\approx 0.21+0.73+0.11=1.05,$$

and the total cost of path 1 solution = $c^{1,2}_{i,j}+c^{2,3}_{j,j}+c^{3,4}_{j,j}+c^{4,6}_{j,k}=2.07$.

Along the same physical path TP1, another possible map-matching solution (TP2) is shown in Figs. 4(b) and 6(b), which is slightly different from TP1 shown in Figs. 4(a) and 6(a) by matching the GPS point at timestamp 2 to the intermediate point along link (i,j), instead of node j. The straight link distance in the former case is much shorter than that in the latter case, while all the other arc costs are the same for both solutions. The updated arc cost $c^{1,3}_{i,j}=d(i,t=1)+\bar{d}^{1,3}_{j,k}\left(g^2_x,g^2_y\right)+d(j,t=3)=0.11+1.8+0.14=2.05, c^{3,4}_{j,j}$ and $c^{4,6}_{j,k}$ are kept the same as the solution TP1. As shown in Fig. 6 (b), the total path cost of TP2 is approximately 2.05 + 0.35 + 1.05 = 3.45, which is higher than 2.07 in Fig. 6(a). This means that the solution in Figs. 4(a) and 6(a) is clearly a better map-matching solution.

On the other hand, when the modeling and sampling intervals are different and there is no location data at time t, we can consider a default value of $d(i,t)=\omega$ in the generalized cost function, where ω is a constant. One can use an approach of interpolating between successive points to come up with a reasonable value of ω. By doing so, we hope to still use $\sum_{i\in \text{Path}}\sum_{t=1}^{T}d(i,t)$ to find the most likely path based on (possibly limited) information on available location data and the underlying network connectivity constraints.

For the GPS trace and network in Fig. 2, if we assume there are no location data at time stamps 2, 3 and 4 in Fig. 7a, we also can use Eq. (5) to calculate the vertexes' distance.X

$$d(i,t=1)=\sqrt{(1-1.1)^2+(1-1.05)^2}\approx 0.11.$$

$$d(j,t=2)=\omega,$$

$$d(j,t=3)=\omega,$$

$$d(j,t=4)=\omega,$$

$$d(k,t=6)=\sqrt{(5-4.95)^2+(5-4.9)^2}\approx 0.11.$$

$$\bar{d}^{4,6}_{j,k}\left(g^5_x,g^5_y\right)=\sqrt{\left[\frac{5-4}{6-4}\times(x_k-x_j)+x_j-4.8\right]^2+\left[\frac{5-4}{6-4}\times(y_k-y_j)+y_j-2.3\right]^2}\approx 0.73.$$

Therefore, assumed $\omega=0$, all arcs cost can be computed along TP1 as plotted in Fig. 7b.

$$c^{1,2}_{i,j}=d(i,1)+d(j,2)\approx 0.11+\omega=0.11,$$

$$c^{2,3}_{j,j}=d(j,2)+d(j,t=3)\approx \omega+\omega=0,$$

$$c^{3,4}_{j,j}=d(j,t=3)+d(j,t=4)\approx \omega+\omega=0,$$

$$c^{4,6}_{j,k}=d(j,t=4)+\bar{d}^{4,6}_{j,k}\left(g^5_x,g^5_y\right)+d(k,t=6)\approx 0.73+0.11=0.84,$$

and the total path cost = $c^{1,2}_{i,j}+c^{2,3}_{j,j}+c^{3,4}_{j,j}+c^{4,6}_{j,k}=0.95$.

6. Solution algorithm

6.1. Time-dependent least cost algorithm using dynamic programming

Using the above mathematical programming model, one can derive the optimality condition, based on Bellman's principle within a dynamic programming (DP) framework. Specifically, $\lambda_{j,t}$ denotes the total travel cost of the current least-cost path from origin r at time 1 to node j at time t. An optimal path in the time-expanded network should satisfy the following conditions:

$$\lambda_{j,t'}=\begin{cases}0, & \text{if } j=r,\ t'=1\\ \min\left(\lambda_{j,t'-1}+c^{t'-1,t'}_{j,j},\lambda_{i,t}+c^{t,t'}_{i,j}\right), & \text{otherwise for feasible}(i,j)\end{cases} \tag{11}$$

We now solve this problem by enhancing label correcting or DP-based algorithms (Ziliaskopoulos and Mahmassani, 1993; Chabini, 1998). It is worth noting that, the proposed DP-based framework does not need to explicitly construct space–time networks for each trace, and the major key steps are label checking and updating.

Algorithm 1 (*offline network time path estimation*).

Input: network G, origin node r, destination node s, and location data from time 1 to T, maximum allowed location-to-node distance μ (e.g. using maximum GPS error range)
Output: The most likely network–time path from $r(1)$ to $s(T)$
Step 1. (Preprocessing)
For each node within the search region, use Eq. (7) to calculate time-indexed location-to-link distance $d(i,t)$ for each time t
Step 2. (Initialization)
$\lambda_{j,t}=\infty,\forall j;\ \lambda_{r,1}=0;$
Step 3. (Label updating)
For time t = 1 to T
For each link (i,j) within the search region at time t
Step 3.1//traveling arc
For time $t'=t$ + FFTT(i,j) to t + MaxTT(i,j)
calculate $c^{t,t'}_{i,j}$ based on Eq. (5)
If $(d(i,t)\leqslant\mu$ and $d(i,t')\leqslant\mu$ and $\lambda_{i,t}+c^{t,t'}_{i,j}\leqslant\lambda_{i,t'})$ **Then**
Update $\lambda_{j,t'}=\lambda_{i,t}+c^{t,t'}_{i,j}$ along traveling arc, and update the corresponding predecessor at vertex $j(t')$ as vertex $i(t)$
End if
End for

(*continued on next page*)

Step 3.2// waiting arc
If (node i allows waiting and $\lambda_{i,t} + c_{i,i}^{t,t+1} \leqslant \lambda_{i,t+1}$) **Then**
Update $\lambda_{i,t+1} = \lambda_{i,t} + c_{i,i}^{t,t+1}$ along dwell arc, update corresponding predecessor at vertex $i(t+1)$ as vertex $i(t)$
End if
End for each link
End for each time
Step 4. (Fetch complete time-dependent likely path) Back-tracing predecessors from super sink vertex $s(T)$ up to super source $r(1)$, and obtain the map-matching result

The flow balance constraints presented in section 3 is inherently embedded in the space–time search process in Step 3, that is, for each time t, for each link (i, j) and for each time t'. Specifically, the flow balance constraints at the origin and destination are satisfied when we start searching from origin $r(1)$, and back-trace from super sink $s(T)$.

Algorithm 2 (*Rolling horizon implementation for real-time estimation*).

Input: network G, origin node r, estimated traffic states up to time $t1$, newly received online location data from time $t1$ to $t2$, maximum allowed location-to-node distance
Output: Likely current location s^* at time $t2$, the most likely network–time path from $r(1)$ to $s(t2)$
Step 1. (Preprocessing)
For each node within the search region, use Eq. (7) to calculate time-indexed location-to-link distance $d(i, t)$ from $t = t1$ to $t2$
Step 2. (Initialization)
$\lambda_{j,t} = \infty, \forall j$, from $t = t1$ to $t2$;
Step 3. (Label updating)
For time $t = t1'$ to $t2$, where $t1' = t1 - q$ and q is the time lag length
For each link (i, j) within the search region at time t
Perform time-dependent label checking and updating, similar to Steps 3.1 and 3.2 in Algorithm 1
End for each link
End for each time
Step 4. (**Fetch partial likely path up to *t2***) Find the most likely current location s^* at time $t2$ as the node with the minimal label cost $\lambda_{i,t2}$ across all nodes i at time $t2$

Back-trace predecessors from the likely current location s^* to t', and obtain the map-matching result, where $t' < t1$, but t' can be later than the original starting time $t = 1$, if a short look-up window is used.

Algorithm 1 can find global optimum solutions and is suitable for offline processing by taking the entire GPS trace into consideration. Essentially, our proposed DP algorithm is able to solve a multi-stage decision problem by breaking it down into a collection of simpler sub-problems. As illustrated in Fig. 8, in Algorithm 2 for real time map matching applications, the developed DP method seeks to solve each on-line traffic state subproblem only once for each stage k, (e.g. 10 min) given that the solutions from previous time stages have been computed and stored. When new data of a new stage (say 8:00 to 8:10 AM) is received from time $t1$ to $t2$, we need to roll back to an earlier time stamp $t1$' (say 7:30 AM) to take into account that a link might take a maximal duration of q intervals to traverse.

Overall, the average-case time complexity of Algorithm 1 is $O(T \times \alpha \times \beta)$, where T is the planning horizon (T is the number of time intervals in the expanded network for space–time path estimation), α is the number of links covered at each time interval within a search region; and β is MaxTT-FFTT for each link. The complexity of Algorithm 2 is reduced to $O(H \times \alpha \times \beta)$, where $H = t2 - t1 + q$ which is the sum of estimation time stage length. In the worst case, the search process has to reach all links, that is, α = m in the above formula. On the other hand, the space complexity of the proposed algorithm 1 is determined by (1) the size of label cost vector $\lambda_{j,t}$, as $n \times T$, and (2) the size of quadratic cost matrix $c_{i,j}^{t,t'}$ with a dimension of $n^2 \times T^2$. For an efficient implementation of the quadratic cost matrix $c_{i,j}^{t,t'}$, one can use hash tables to store feasible values in $c_{i,j}^{t,t'}$ for each physical link and possible t-to-t' pairs, as the number of physical links is dramatically less than the number of links (i.e.,n^2) in the complete graph, and the average out degree of a node in a transportation is between 2 and 4.

6.2. Handling dwell and detour time at intermediate destinations

The identified time-dependent trajectory is extremely useful for identifying a trip chain that contains intermediate stops and detours. Fig. 9 illustrates how the proposed algorithm represents the detour based on the 4-node and 5-link road network with 7 GPS points. In this example, GPS point set is = {1, 2, 3, 4, 5, 6, 7}, and GPS curve set is= {1 → 2, 2 → 3, 3 → 4, 4 →

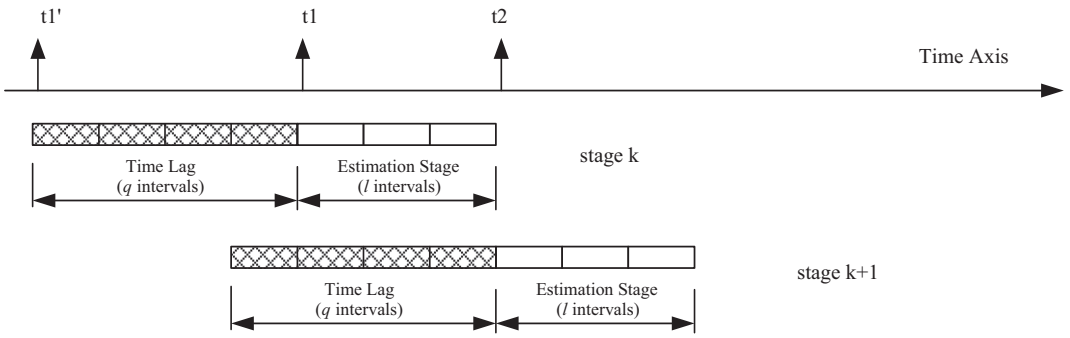


Fig. 8. Algorithm 2: Rolling horizon implementation for real-time space–time path estimation.

5, 5 → 6, 6 → 7}. A traveler goes to shopping mall 1 from his/her home, and then coming back before driving to shopping mall 2. This leads a path with loop $i1 \rightarrow j \rightarrow i1 \rightarrow i2 \rightarrow k$.

For a systematic comparison, we can match this GPS trace to a 4-node and 5-link road network based on the different algorithms.

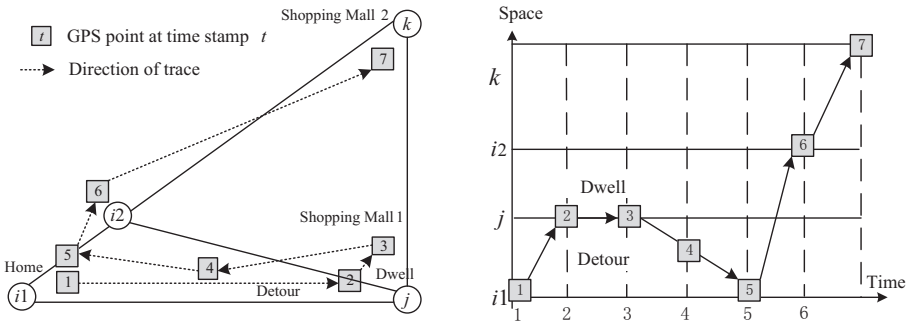
(i) A point-to-point algorithm could produce a solution where GPS point 1 matches node $i1$, point 2 matches node j, point 3 matches node j, points 4, 5 and 6 match node $i2$, and point 7 matches node k. As a result, the matched link set is = $\{i1 \rightarrow j, j \rightarrow i2, i2 \rightarrow k\}$;

(ii) Based on point-to-link algorithm, GPS points 1 and 2 match link $i1 \rightarrow j$, point 3 matches link $j \rightarrow k$, point 4 matches link $j \rightarrow i2$, point 5 matches link $i1 \rightarrow i2$, and points 6 and 7 match link $i2 \rightarrow k$. Similarly, matched link set is = $\{i1 \rightarrow j, j \rightarrow k, j \rightarrow i2, i1 \rightarrow i2, i2 \rightarrow k\}$;

(iii) A curve-to-link algorithm can generate that GPS curve 1 → 2 matches link $i1 \rightarrow j$, curve 2 → 3 matches link $j \rightarrow k$, curves 3 → 4 and 4 → 5 match link $j \rightarrow i2$, curve 5 → 6 matches link $i1 \rightarrow i2$, and curve 6 → 7 matches link $i2 \rightarrow k$, which leads to matched link set as $\{i1 \rightarrow j, j \rightarrow k, j \rightarrow i2, i1 \rightarrow i2, i2 \rightarrow k\}$;

(iv) Based on time-dependent least cost algorithm using our proposed dynamic programming the matched result is time dependent path as shown in Fig. 9b, namely $i1(1) \rightarrow j(2) \rightarrow j(3) \rightarrow i1(5) \rightarrow i2(6) \rightarrow k(7)$, with a matched link set as = $\{i1 \rightarrow j, j \rightarrow i1, i1 \rightarrow i2, i2 \rightarrow k\}$.

Accordingly, if only the geometric distances from GPS points 1, 2, 3, 4 and 5 to link in the network are used in a map-matching algorithm, then it is not trivial to decide if and how the detour should be allowed. Recall that standard label correcting and setting algorithms do not allow tours with loops. However, using the space–time representation, the resulting space–time path is still a simple path without loops through time-indexed vertices, as shown in Fig. 9b. Accordingly, a dwell arc and a detour arc can be automatically identified with associated activity time duration. Such information on trip chaining can greatly help to examine sample travelers' movements throughout the day with different trip purposes and dwell times at different activity locations.



(a) Hypothetical 4-node network based on two-dimensional network for the detour issue

(b) Space-time path in time -expanded network representations for the detour issue

Fig. 9. An illustrative example with a detour and a dwell activity.

6.3. Quantifying estimation uncertainty based on network potential path area

We now calculate the uncertainty associated with an estimated result by adopting the concept of the Potential Path Area (PPA) within an underlying network. First, we drive the least cost path trees based on the proposed network time path estimation algorithm, one from the origin and the other one towards the destination, which lead to the forward optimal label cost $\lambda^F_{i,t}$ from the origin to vertex (i, t) and the backward optimal label cost $\lambda^B_{i,t}$ from vertex (i, t) to the destination.

Secondly, given the minimum estimated cost C^* and the error bound ε, the time budget for traveling can be determined by the maximum tolerant cost $(C^* + \varepsilon)$. Accordingly, the PPA can be defined to delimit all accessible vertices. Since a location within the network is accessible if and only if its forward and backward optimal label costs together do not exceed the time budget. The set of all accessible vertices can be defined analytically as:

$$M = \left\{(i,t) \in V | \lambda^F_{i,t} + \lambda^B_{i,t} < C^* + \varepsilon\right\} \tag{12}$$

Which is subject to the error bound ε that reflects the tolerant level. For example, the error bound ε can be calculated as a function of the number of samples points times the average estimation error say 0.5 m. Finally, we use the set M to represent the size of PPA within the network subject to estimation quality constraints. Hence, the total number of vertices M describes the uncertainty or confident level associated with the estimation result: more accessible vertices provide more feasible arcs that a trajectory can be matched to, which results in higher uncertainty associated with an estimation result. By doing so, we can use the numbers of possible space–time vertices as a proxy to evaluate the uncertainty changes under different data collection settings. This sensitivity analysis result can provide critical information for selecting different sampling rates and additional information sources such as AVI, AVL at various locations.

In an extreme case, there are only two accessible vertices, which means that a series of GPS points can be matched only to that arc; therefore, there is no uncertainty associated with the result. Fig. 10 shows general cases where there are multiple feasible paths. The introduction of uncertainty based on the network PPA can provide additional information rather than as set of most likely map-matching space–time paths, and it may also reduce the required sample interval (e.g. second-by-second). For instance, Fig. 10(b) has larger PPA than Fig. 10(a), which includes two more vertices and result in one more possible feasible path. Therefore, even the map-matching results are the same, the first result is 100% sure that the result is correct, while the second has an uncertainty associate with the estimation result because that are other feasible options available. Besides, comparison of Fig. 10(a) and (b) show that the network PPA can reduce the sampling size (e.g. 1 s to 2 s).

7. Numerical experiments

The applicability of proposed methods is illustrated using empirical data from three U.S. cities. We evaluate our algorithm using four network–time path estimation metrics:

(i) Computational time (Ct):

$$Ct = \frac{Total\ caculation\ time}{Number\ of\ GPS\ trajectories} \tag{13}$$

(ii) Average link identification rate (Al):

$$Al = \frac{Number\ of\ correctly\ matched\ road\ links}{Number\ of\ matched\ links\ for\ a\ traveller} \tag{14}$$

(iii) Average travel time errors for correctly identified links (Ae):

$$Ae = \frac{\sum Absolute\ travel\ time\ error\ of\ correctly\ matched\ links}{\sum Actual\ travel\ time\ of\ all\ correctly\ matched\ links} \tag{15}$$

(iv) Average relative travel time error (Ar):

$$Ar = \frac{Ae}{\sum average\ travel\ time\ of\ links} \tag{16}$$

To illustrate the above measures, Table 6 shows estimated results for a sample GPS trace in the hypothetic network in Fig. 2b. As correctly matched road links are links $i \to j$, $j \to k$, the average link identification rate $Al = 2/4 = 50\%$. At last, $Ae = (0 + 2)/2 = 1$, according to Eq. (15).

We implement the DP algorithm using Visual C# 2010. We performed all of the experiments on a Lenovo ThinkPad E40 laptop with 2.53 GHz Intel i5 CPU and 4 GB memory. First, the C# algorithm is tested against a linear programming model implemented by GNU Linear Programming Kit (GLPK, 2012), and the latter open-source Linear Programming solver produces the same results but with much longer running time. We then tested the performance of the map-matching algorithm on three networks, namely New York City (New York), Salt Lake City (Utah) and Phoenix (Arizona). Table 7 shows their basic

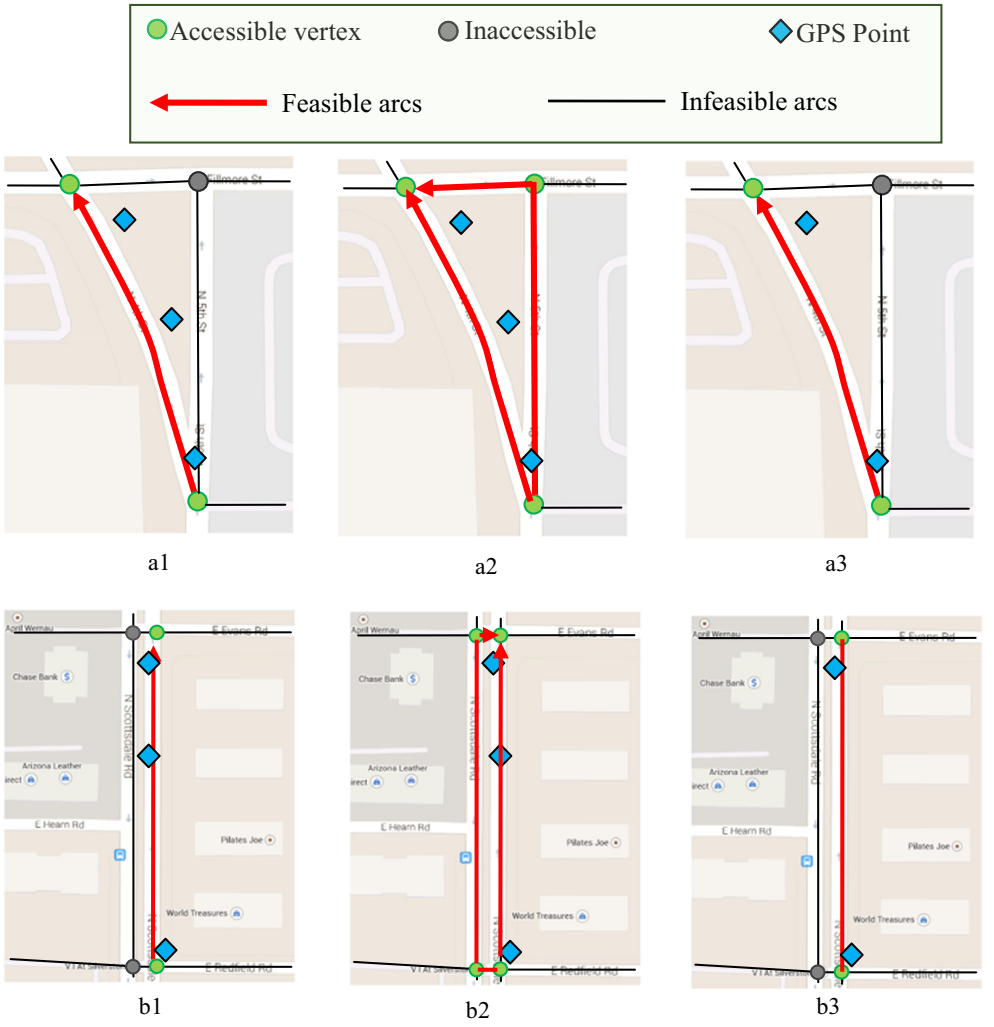


Fig. 10. Illustrative of quantifying estimating uncertainty.

Table 6
Illustration for network–time path estimation quality measures in hypothetic network.

Constructed ground truth path (link id)	$i \rightarrow j$	$j \rightarrow k$	$k \rightarrow j$	$j \rightarrow i$
Estimate link of path	$i \rightarrow j$	$j \rightarrow k$	$j \rightarrow k$	$i \rightarrow j$
Constructed ground truth time stamps (second)	4	2	3	4
Estimate travel time	4	4	2	3
Link identification error			Incorrect	Incorrect
Average distance (meter)	5	5	5	5
Absolute travel time estimation error for correctly identified travel time	0	2	*N/A*	*N/A*

network attributes and Fig. 11 illustrates the networks, We generate the New York City GPS data through a mesoscopic dynamic traffic assignment simulator, and a small percentage of intermediate stops are generated at various locations. The last two GPS data sets in Salt Lake City and Phoenix contain 2 and 43 real-world GPS data traces, and externally collected travel time records on links, in July 2013 and April 2014 respectively. The first synthesized data set in New York City with a large number of GPS traces serve as the ground truth data for evaluating the impact of GPS data error range, while the last two data sets are used to identify potential issues raised in a real-world setting.

For experiments on the New York network with synthesized data, we first assume the error of GPS points follow a normal distribution, with a mean of $\mu = 0$ and standard deviations as $\sigma = 10$ and 50 m, respectively. As the probability Prob($\mu - 3\sigma$, $\mu + 3\sigma$) = 99.7% for a normal distribution, we set the maximum error range parameter σ (previously defined for reducing prob-

Table 7
Three transportation networks attributes.

Network	#of nodes	#of links	#of traces	Avg. number of second-by-second samples per trace
New York City	9390	21,734	1200 (synthesized)	865 sample points
Salt Lake City	21,381	50,893	2	1499 sample points
Phoenix	19,523	47,713	43	926 sample points

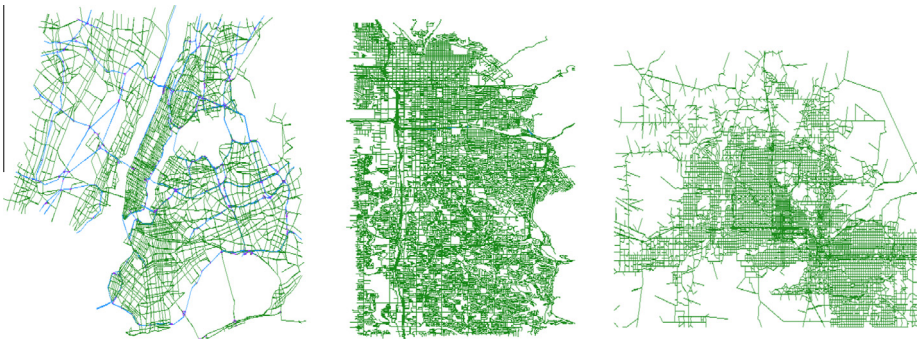


Fig. 11. New York City, Salt Lake City and Phoenix road networks (from left to right).

lem space in Section 5.3) as 4σ. Table 8 examines the map-matching quality for the simulated datasets. When the actual error standard deviation σ is reasonably small such as 10 m, it is easy to obtain very satisfactory results with close to 100% identification rates and low running time (1.39 s per trace). Given a large GPS error of Standard Deviation σ = 50 m, we observe an increase in computational time due to the extended search range. The percentage of trajectories with all links being correctly identified is downgraded to 99.5%. Overall, when a trace contains a large number of noisy GPS points on a high-density network, the running time of this approach increases dramatically, for example, reaching 2.77 s per trace in the New York City network with assumed standard deviation σ = 50 m.

For the last two data sets with real-world GPS traces, we collected 2 and 43 traces which pass different link types including highway, arterial, collector, frontage road and ramp. As shown in Table 8, the network–time path estimation results still reach a satisfactory level with link identification rates of 98.2%, 98.9%, 98.8% and 99.3% under different search regions and networks. The obtained correct link identification rate of 99.3% in Phoenix network is similar to the previously reported performance (see Quddus et al., 2003; Quddus, 2006; Quddus et al., 2007; Velaga et al., 2009).

In order to compare with commonly used existing algorithms, we also implemented GPS map matching methods with different distance measures, within a generic modeling framework as shown in Appendix A. Under the Phoenix network with one second frequency, we obtained 65% and 88% and 92% as the rates of correctly identified links, respectively, based on the commonly-used distance measures, namely point-to-point, point-to-curve and curve-to-curve methods. In comparison, the proposed optimization model in the time-expanded network can systematically capture the distance between a sequence of GPS points and a subpath of links in a transportation network. That is, the existing methods typically focus on the distance between individual objects (such as GPS curve, link, or node), while our method can take into account a wide range of distance measures between a large number of GPS points and a dynamically defined subset of links. With the help of quadratic distance/cost matrix,$c_{i,j}^{t,t'}$, our method is able to further identify complex space–time activity patterns such as detours, intersection stops and activity stops.

As the network structure is taken into account automatically in the least cost path search algorithm, the proposed algorithm can inherently maintain connectivity through links along a path, and it also shows superior performance in precisely recognizing the distance from GPS points to a set of road segments in an optimization-driven framework. It should be noticed that, if one only considers simply the distance from GPS points to a single node or a single link, then it could fail to capture several major factors in real-world trajectories, such as the link-to-link connectivity, possibility of dwelling and stationary activities.

The link identification rate of the two real-world data sets is relatively lower than the rate based on simulated data sets, and in particular, the computational time per trace reaches 5.39 s in Salt Lake City network with 200-m search region, as the underlying structure has a very high fidelity node-link coverage. There are few potential reasons for possible mismatches when applying the proposed algorithm in real-world settings. The proposed algorithm explicitly allows loops around nodes, so it introduces possibilities of visiting a node more than once. This issue is more likely to occur when adjacent links of a node are very short, especially compared to the GPS error range defined by σ in our algorithm. The challenge associated with a real-world data set is that, the GPS error range is a time-varying and situation-dependent parameter, so estimating are

Table 8
Experimental results with 1-s sampling time interval.

Road network	Error standard deviation σ (m)	Search region μ (m)	Computational time (Ct) per trace (s)	Link identification rate (Al) (%)	Average absolute travel time error per link (Ae) (s)	Average relative travel time error (%)	Computational time (Ct) per stage(30-s) of trace (s)
New York	10	40	1.39	99.9	3.2	2.2	0.048
New York	50	200	2.77	99.5	4.7	11.4	0.096
Salt Lake City	–	40	2.91	98.2	5.5	7.3	0.058
Salt Lake City	–	200	5.39	98.9	4.6	12.7	0.109
Phoenix	–	40	1.66	98.8	4.1	10.3	0.056
Phoenix	–	200	3.08	99.3	3.5	4.8	0.100

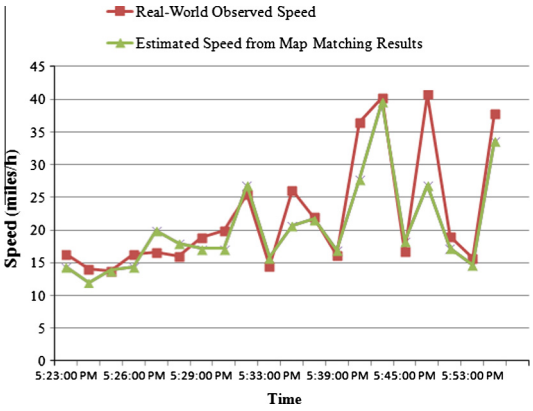


Fig. 12. Comparison between real-world speed and map-matched results (1 s sampling interval).

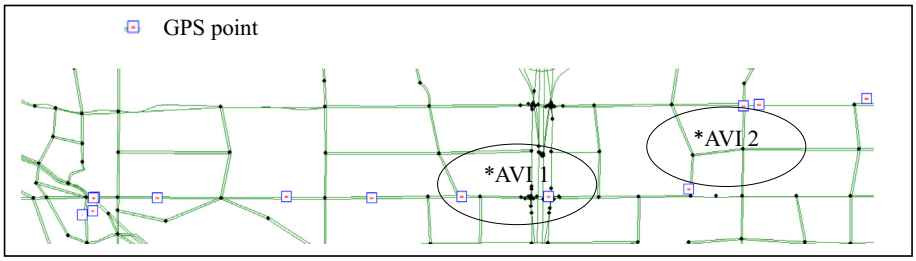


Fig. 13. Phoenix Subarea network and 12 raw GPS points with 60-s sampling time interval.

liable GPS error range is quite important for the proposed algorithm. This type of error can be mitigated by adding a reasonable minimum time threshold for revisiting a node. Also note that the maximum computational time of per stage of trace (30-s) is 0.109 s, therefore, our rolling horizon implementation can be applied to real-time nature of the traffic estimation applications.

Table 8 gives a range of the average relative travel time error between 2.2% and 12.7%. Specifically, Fig. 12 shows the comparison results between the estimated speeds and the actual speed from the Salt Lake City data set, while the actual speed data are calculated based on the directly measured travel time records by non-driver data collectors using high-solution clocks and GPS devices. The figure shows that the error is relatively small between estimated and real-world values when the driving speed is low (corresponding to a long link travel time). The large speed discrepancy is typically observed on freeway links with long travel time on mainline segments, as well as complex cases with short and complex ramps on both ends.

An important practical issue for the network–time path estimation is how to handle relatively long GPS sampling time interval, e.g., with a long range of 5–180 s. In this case, as illustrated in Fig. 13, there are multiple subpaths between two

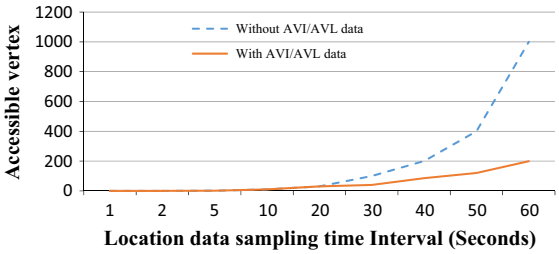


Fig. 14. Number of possible space–time vertices under different location data sampling rates without and with AVI and additional AVL data based on PPA.

consecutive sample locations. Therefore, when we reduce the sample frequency (e.g. more than 30 s), the number of possible matched vertices will quickly increase. On the other hand, we find that the number of uncertainty subpaths depending on the sampling time interval and the density of road network. At high density of subarea, it could reduce uncertainty based on AVI and additional AVL data, e.g. high-frequency Differential Global Positioning System (DGPS) data, from other detectors. As Fig. 10 illustrates, we use PPA to delimit all feasible routes in space, and obtain all accessible vertices. Consequently, the numbers of possible space–time vertices under different location data sampling rates without and with AVI and AVL data in Phoenix Subarea network are shown in Fig. 14. The proposed optimization method still allows us to systematically select not only highly possible map-matched links and but also the most likely sub-paths among several candidate alternatives.

For 60-s low frequency GPS data, the correct identification rates of the algorithm are 97.8% without additional data, which is similar to the performance reported by a recent study by Quddus and Washington (2015). If AVI or AVL data could be used at high density subarea network, the algorithm further provides a 99.2% correct link identification.

8. Conclusion

This paper develops a novel path-oriented traffic state estimation model based on time geographic principles and a time-expanded network representation. We adapt a dynamic programming algorithm to solve the proposed models. The proposed algorithm cannot only find the most likely used road path, but also estimate the resulting link travel times and activity duration at possible intermediate stops, as well as the estimation uncertainty measured by the number of accessible space–time vertices. The optimal solution algorithms are applicable for both offline data mining applications and real-time traffic estimation tasks.

While focusing on improvements in estimation accuracy, this paper also develops a rigorous optimization framework that can systematically utilize both spatial and temporal distance measures and network connectivity. Compared to the existing point-to-point, point-to-link and curve-to-curve GPS map matching algorithms, our algorithm can achieve similar accuracy when the commonly used spatial distance measures perform well with high sampling rates and relatively simple topology. Under low sample rates and complex temporal and spatial activities, our proposed algorithm can simultaneously estimate the travel time and traveled links within the network connectivity constraints to find the likely paths and further quantify the uncertainty level of the estimation results.

Future research will be focused on how to handle possible GPS trajectories with low sampling rates without another source data and resulting large spatial error range. In the future study, with the proposed time-expanded graph modeling framework, we will further consider how to (i) systematically take into account the interactions between multiple vehicle trajectories, e.g., using a dynamic time warping method proposed by Taylor et al. (2015) for studying intradriver heterogeneity; (ii) how to feed a large number of map-matched GPS or cell phone traces to a simulation based dynamic traffic assignment engine, e.g., DTALite proposed by Zhou and Taylor (2014), for rapid traffic congestion prediction. For real-time applications, we should systematically evaluate the performance and accuracy of Algorithm 2 within a rolling horizon framework with different degrees of sample rates, rolling stage length, and complexity of underlying network topologies.

Acknowledgements

The material in this paper is based on research supported by National Science Foundation – United States under Grant No. BCS-1224102 "Measuring the Environmental Costs of Space–time Prisms in Sustainable Transportation Planning".

Appendix A. Simple time-invariant path search algorithms using existing GPS distance methods

Many studies have implemented the point-to-point, point-to-curve and curve-to-curve geometric algorithms. In this paper, we adapt their distance functions to test our experiments. We use λ_j (without temporal dimension) denote the total

travel cost of the current least-cost path from origin r to node j. Define search region SR(i,j) as the set of GPS points within a pre-set space distance around link (i,j).

Input: network G, origin node r, destination node s, location data and maximum allowed location-to-node distance μ (e.g. using maximum GPS error range)

Output: The most likely set of link from $r(1)$ to $s(T)$

Step 1. (Preprocessing)

For each node within the search region, calculate

(i) location-to-node distance $d(i,t)$ for each GPS record GPS point (g_x^t, g_y^t) to node i.

(ii) location-to-link distance $d(i,j,t)$ for each GPS point (g_x^t, g_y^t) to link $i \to j$

(iii) curve-to-link distance $d(i,j,t,t+1)$ for consecutive 2 GPS records, that is, $(g_x^t, g_y^t) \to (g_x^{t+1}, g_y^{t+1})$ to link $i \to j$.

Step 2. (Initialization)

$\lambda_j = \infty \forall j;\ \lambda_r = 0;$

Define time-invariant link cost $c_{i,j}$ as the following for each link (i,j)

(i) $c_{i,j} = \min_{t \in SR(i,j)} d(i,t)$ for point-to-point distance

(ii) $c_{i,j} = \min_{t \in SR(i,j)} d(i,j,t)$ for point-to-link distance

(iii) $c_{i,j} = \min_{t \in SR(i,j)} d(i,j,t,t+1)$ for point-to-curve distance

Step 3. (Perform label updating in label correcting or label setting algorithm)

For each link (i,j) in the scan eligible list

If $(\lambda_i + c_{i,j} \leqslant \lambda_j)$ **Then**

Update $\lambda_j = \lambda_i + c_{i,j}$ and update the corresponding predecessor

End if

End for

Step 4. (Fetch complete time-invariant likely path) Back-tracing predecessors from super sink vertex s up to super source r, and obtain the map-matching result

---

References

Ahuja, R.K., Magnanti, T.L., Orlin, J.B., 1993. Network Flows: Theory, Algorithms, and Applications.

Bernstein, D., Kornhauser, A., 1998. An Introduction to Map Matching for Personal Navigation Assistants.

Chabini, I., 1998. Discrete dynamic shortest path problems in transportation applications: complexity and algorithms with optimal run time. Transp. Res. Rec. 1645, 170–175.

Cremer, M., Papageorgiou, M., 1981. Parameter identification for a traffic flow model. Automatica 17 (81), 837–843.

Deng, W., Lei, H., Zhou, X., 2013. Traffic state estimation and uncertainty quantification based on heterogeneous data sources: a three detector approach. Transp. Res. Part B: Meth. 57, 132–157.

Downs, J.A., Horner, M.W., 2012. Probabilistic potential path trees for visualizing and analyzing vehicle tracking data. J. Transp. Geogr. 23, 72–80.

Fu, M., Li, J., Wang, M., 2004. A hybrid map matching algorithm based on fuzzy comprehensive judgment. In: Paper Presented at International IEEE Conference on Intelligent Transportation Systems.

GLPK, 2012. GNU Linear Programming Kit. Available at <http://www.gnu.org/software/glpk>.

Greenfeld, J.S., 2002. Matching GPS observations to locations on a digital map. In: Paper Presented at Transportation Research Board 81st Annual Meeting.

Hägerstrand, T., 1970. What about people in Regional Science? Pap. Reg. Sci. Assoc. 24 (1), 6–21.

Herrera, J.C., Bayen, A.M., 2010. Incorporation of lagrangian measurements in freeway traffic state estimation. Transp. Res. Part B: Meth. 44 (4), 460–481.

Honey, S.K., Loughmiller Jr., G.E., Milnes, K.A., Phillips, A.C., White Jr., M.S., Zavoli, W.B., 1989. Vehicle Navigational System and Method.

Jo, K., Chu, K., Sunwoo, M., 2012. Interacting multiple model filter-based sensor fusion of GPS with in-vehicle sensors for real-time vehicle positioning. IEEE Trans. Intell. Transp. Syst. 13 (1), 329–343.

Kim, W., Jee, G-I., Lee, J., 2000. Efficient use of digital road map in various positioning for ITS. In: Paper Presented at Position Location and Navigation Symposium, IEEE 2000.

Kong, Q., Zhao, Q., Wei, C., Liu, Y., 2013. Efficient traffic state estimation for large-scale urban road networks. IEEE Trans. Intell. Transp. Syst. 14 (1), 398–407.

Krakiwsky, E.J., Harris, C.B., Wong, R.V.C., 1988. A Kalman filter for integrating dead reckoning, map matching and GPS positioning. In: Paper Presented at Position Location and Navigation Symposium, 1988. Record. Navigation into the 21st Century. IEEE PLANS'88, IEEE.

Kuijpers, B., Othman, W., 2009. Modeling uncertainty of moving objects on road networks via space-time prisms. Int. J. Geograph. Inf. Sci. 23 (9), 1095–1117.

Kwan, M.P., 1998. Space-time and integral measures of individual accessibility: a comparative analysis using a point-based framework. Geograph. Anal. 30 (3), 191–216.

Meng, Y., 2006. Improved Positioning of Land Vehicle in ITS Using Digital Map and Other Accessory Information (Ph.D. thesis). Department of Land Surveying and Geoinformatics, Hong Kong Polytechnic University.

Miller, H.J., 1991. Modelling accessibility using space-time prism concepts within geographical information systems. Int. J. Geograph. Inf. Syst. 5 (3), 287–301.

Muñoz, L., Sun, X., Horowitz, R., Alvarez, L., 2003. Traffic density estimation with the cell transmission model. In: Paper Presented at American Control Conference, 2003. Proceedings of the 2003.

Obradovic, D., Lenz, H., Schupfner, M., 2006. Fusion of map and sensor data in a modern car navigation system. J. VLSI Signal Process. Syst. Signal Image Video Technol. 45 (1–2), 111–122.

O'Sullivan, D., Morrison, A., Shearer, J., 2000. Using desktop GIS for the investigation of accessibility by public transport: an isochrones approach. Int. J. Geograph. Inf. Sci. 14 (11), 85–104.

Peker, A.U., Tosun, O., Acarman, T., 2011. Particle filter vehicle localization and map-matching using map topology. In: IEEE Intelligent Vehicles Symposium (IV). pp. 248–253.

Pyo, J-S, Shin, D-H., Sung, T-K., 2001. Development of a map matching method using the multiple hypothesis technique. In: Paper Presented at Intelligent Transportation Systems, 2001. Proceedings. 2001 IEEE.

Quddus, M.A., 2006. High Integrity Map Matching Algorithms for Advanced Transport Telematics Applications.

Quddus, M., Washington, S., 2015. Shortest path and vehicle trajectory aided map-matching for low frequency GPS data. Transp. Res. Part C: Emerg. Technol. 55, 328–355.
Quddus, M.A., Ochieng, W.Y., Zhao, L., Noland, R.B., 2003. A general map matching algorithm for transport telematics applications. GPS Solut. 7 (3), 157–167.
Quddus, M.A., Ochieng, W.Y., Noland, R.B., 2007. Current map-matching algorithms for transport applications: State-of-the art and future research directions. Transp. Res. Part C: Emerg. Technol. 15 (5), 312–328.
Sun, X., Muñoz, L., Horowitz, R., 2003. Highway traffic state estimation using improved mixture Kalman filters for effective ramp metering control. In: 2003 Proceedings 42nd IEEE Conference on Paper Presented at Decision and Control.
Syed, S., Cannon, M.E., 2004. Fuzzy logic-based map matching algorithm for vehicle navigation system in urban canyons. In: Paper presented at ION National Technical Meeting, San Diego, CA.
Szeto, M.W., Gazis, D.C., 1972. Application of Kalman filtering to the surveillance and control of traffic systems. Transp. Sci. 6 (4), 419–439.
Taylor, J., Zhou, X., Rouphail, N., Porter, R.J., 2015. Method for investigating intradriver heterogeneity using vehicle trajectory data: a dynamic time warping approach. Transp. Res. Part B: Meth. 73, 59–80.
Thiagarajan, A., Ravindranath, L., Balakrishnan, H., Madden, S., Girod, L., 2011. Accurate, Low-Energy Trajectory Mapping for Mobile Devices. NSDI.
Velaga, N.R., Quddus, M.A., Bristow, A.L., 2009. Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems. Transp. Res. Part C: Emerg. Technol. 17 (6), 672–683.
Wang, Y., Papageorgiou, M., 2005. Real-time freeway traffic state estimation based on extended Kalman filter: a general approach. Transp. Res. Part B: Meth. 39 (2), 141–167.
White, C.E., Bernstein, D., Kornhauser, A.L., 2000. Some map matching algorithms for personal navigation assistants. Transp. Res. Part C: Emerg. Technol. 8 (1), 91–108.
Work, D.B., Blandin, S., Tossavainen, Olli-Pekka, Piccoli, B., Bayen, A.M., 2010. A traffic model for velocity data assimilation. Appl. Math. Res. Exp. 1, 1–35.
Yang, L., Zhou, X., 2014. Constraint reformulation and a Lagrangian relaxation-based solution algorithm for a least expected time path problem. Transp. Res. Part B 59 (1), 22–44.
Yang, D., Cai, B., Yuan, Y., 2003. An improved map-matching algorithm used in vehicle navigation system. In: Paper Presented at Intelligent Transportation Systems, 2003. Proceedings. 2003 IEEE.
Yin, H., Wolfson, O., 2004. A weight-based map matching method in moving objects databases. In: 2004 Proceedings 16th International Conference on Paper Presented at Scientific and Statistical Database Management.
Zhao, Y., 1997. Vehicle Location and Navigation System. Artech House, Inc., MA.
Zhou, X., Taylor, J., 2014. DTALite: a queue-based mesoscopic traffic simulator for fast model evaluation and calibration. Cogent Eng. 1, 961345.
Ziliaskopoulos, A., Mahmassani, H.S., 1993. A time-dependent shortest path algorithm for real-time intelligent vehicle/highway systems applications. Transp. Res. Rec. 1408, 94–100.