A generic data assimilation framework for vehicle trajectory reconstruction on signalized urban arterials using particle filters

Xie, Xu; van Lint, Hans; Verbraeck, Alexander

DOI
10.1016/j.trc.2018.05.009

Publication date
2018

Document Version
Final published version

Published in
Transportation Research Part C: Emerging Technologies

Transportation Research Part C 92 (2018) 364–391

Contents lists available at ScienceDirect

Transportation Research Part C

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

A generic data assimilation framework for vehicle trajectory reconstruction on signalized urban arterials using particle filters

Xu Xie[a,c,*], Hans van Lint[b], Alexander Verbraeck[a]

[a] Department of Multi Actor Systems, Faculty of Technology, Policy, and Management, Delft University of Technology, Jaffalaan 5, P.O. Box 5015, 2628 BX Delft, Netherlands

[b] Department of Transport and Planning, Faculty of Civil Engineering and Geosciences, Delft University of Technology, Stevinweg 1, P.O. Box 5048, 2600 GA Delft, Netherlands

[c] Department of Modeling and Simulation, College of System Engineering, National University of Defense Technology, 410073 Changsha, China

ARTICLE INFO

Keywords:
Vehicle trajectory reconstruction
Noisy sensor data
Vehicle accumulation estimation
Microscopic traffic simulation
Data assimilation
Particle filters

ABSTRACT

With trajectory data, a complete microscopic and macroscopic picture of traffic flow operations can be obtained. However, trajectory data are difficult to observe over large spatiotemporal regions—particularly in urban contexts—due to practical, technical and financial constraints. The next best thing is to estimate plausible trajectories from whatever data *are* available. This paper presents a generic data assimilation framework to reconstruct such plausible trajectories on signalized urban arterials using microscopic traffic flow models and data from loops (individual vehicle passages and thus vehicle counts); traffic control data; and (sparse) travel time measurements from whatever source available. The key problem we address is that loops suffer from miss- and over-counts, which result in unbounded errors in vehicle accumulations, rendering trajectory reconstruction highly problematic. Our framework solves this problem in two ways. First, we correct the systematic error in vehicle accumulation by fusing the counts with sparsely available travel times. Second, the proposed framework uses particle filtering and an innovative hierarchical resampling scheme, which effectively integrates over the remaining error distribution, resulting in plausible trajectories. The proposed data assimilation framework is tested and validated using simulated data. Experiments and an extensive sensitivity analysis show that the proposed method is robust to errors both in the model and in the measurements, and provides good estimations for vehicle accumulation and vehicle trajectories with moderate sensor quality. The framework does not impose restrictions on the type of microscopic models used and can be naturally extended to include and estimate additional trajectory attributes such as destination and path, given data are available for assimilation.

1. Introduction

Vehicle trajectory data provide critically important information for many application areas, ranging from calibration and validation of microscopic traffic flow models (Kesting and Treiber, 2008; Punzo and Montanino, 2016), traffic state reconstruction (van Lint and Hoogendoorn, 2010; Wang et al., 2006), travel time estimation (van Lint, 2010; Coifman, 2002), vehicle energy/emissions estimation (Sun et al., 2015; da Rocha et al., 2015), to name just a few. With trajectory data, a complete picture of traffic flow operations can be obtained, both microscopically and macroscopically. Trajectory data can be collected using a wide range of sensing

* Corresponding author at: Department of Multi Actor Systems, Faculty of Technology, Policy, and Management, Delft University of Technology, Jaffalaan 5, P.O. Box 5015, 2628 BX Delft, Netherlands.
E-mail addresses: x.xie@hotmail.com (X. Xie), J.W.C.vanLint@tudelft.nl (H. van Lint), A.Verbraeck@tudelft.nl (A. Verbraeck).

https://doi.org/10.1016/j.trc.2018.05.009
Received 22 October 2017; Received in revised form 24 March 2018; Accepted 9 May 2018
Available online 22 May 2018
0968-090X/

technologies, such as aerial photography, video, and mobile traffic sensors based on GPS and/or GSM (Montanino and Punzo, 2015; Sun and Ban, 2013). Whereas reconstructing trajectories from microscopic information (e.g., aerial images, GSM traces) requires considerable methodological effort in itself (Montanino and Punzo, 2015), there are only a few cases for which comprehensive data that cover 100% of all vehicle trajectories are available. Collecting such comprehensive trajectory data over large distances (routes, networks) and long time periods is expensive. With infrastructure based sensors or aerial imaging, the collected data can only cover a limited spatiotemporal region due to the high installation and maintenance costs, and it will take a few years (if not longer) before 100% of all vehicles/drivers are equipped with location tracking systems or apps that continuously communicate these data for use in modeling, control or other applications. The next best alternative is to *estimate* vehicle trajectories from whatever data *are* available.

1.1. Estimating vehicle trajectories

Many methods for estimating vehicle trajectories from data have been proposed, which in general terms (should) combine three ingredients: data (from various types of sensors); assumptions (models, equations) that describe the relation between the data and the underlying vehicle trajectories; and data assimilation techniques that combine these ingredients and in the process address measurement and modeling errors. For example, Coifman (2002) reconstructs vehicle trajectories in order to estimate travel times on freeways using traffic data from a single dual loop detector. The proposed method exploits basic traffic flow theory to extrapolate local traffic conditions to an extended freeway link. However, this method will fail when a queue covers the link, which is very common in signalized arterials. An obvious remedy is to reconstruct vehicle trajectories along a route using multiple loop detectors (van Lint, 2010), so that the trajectory reconstruction process is based on information from both up- and downstream locations. The resulting vehicle trajectories are essentially idealized average vehicle trajectories, similar to Lagrangian solutions of kinematic wave theory. However, the adaptive smoothing method used in van Lint (2010) is not suitable for urban trajectory reconstruction in the original form (Treiber and Helbing, 2002; van Lint and Hoogendoorn, 2010), because it would smooth speeds over intersections. Adjusting the method for use in urban settings seems doable in principle, but has not been reported yet. Mehran et al. (2012) propose a data fusion framework to reconstruct vehicle trajectories on urban arterials by incorporating probe and fixed sensor data and the signal timing parameters. The proposed method is also based on kinematic wave theory and employs the variational formulation (VF) (Daganzo, 2005a; Daganzo, 2005b) to solve a dense solution network which is constructed by discretizing the time–space plane. The key principle of the VF method is that the cumulative number of vehicles at each node in this solution network can be computed by a shortest path search from nodes where the cumulative numbers are known (boundary conditions). As a result, any curve which connects nodes with the same cumulative number in the solution network represents an individual vehicle trajectory. Sun and Ban (2013) also apply the VF method to estimate trajectories by fusing probe vehicle trajectories and the signal timing data. However, in both papers using the VF method, sensor errors such as miss-counting and over-counting are not considered; whereas these pose common and difficult problems when using loop detector data (Lu et al., 2008). With these errors, the cumulative numbers at boundaries will be inaccurate. Worse still, the error resulting from using such erroneous counts in the estimation of the number of vehicles between these cumulative count stations becomes unbounded (van Lint and Hoogendoorn, 2015). Another problem is that, due to the use of first order traffic flow theory, the speeds (trajectory slopes) between nodes are piece-wise constant (no acceleration) yielding piecewise linear vehicle trajectories. As shown by Sun et al. (2015) and da Rocha et al. (2015), in case such piece-wise linear trajectories are used to estimate energy consumption or emissions, large errors result, since energy consumption and emissions are influenced largely by the acceleration/deceleration process. This point holds for any traffic state estimation method based on first order traffic flow theory (e.g., Nantes et al., 2016; Yuan et al., 2012; van Hinsbergen et al., 2012), with which (indirect) also vehicle trajectories can be estimated.

In general, when the (behavioral) assumptions of these trajectory estimation methods are insufficient for the application at hand, more elaborate assumptions are required. This could involve estimation methods using higher order macroscopic models (that include speed dynamics as in Wang et al. (2006)), or methods using microscopic models for driving behavior. Goodall et al. (2016) present such a microscopic estimation method for vehicle trajectories on freeways. The objective here is to use trajectory estimation to artificially increase the penetration rate (in sample size as well as frequency) of connected vehicles. The method proposes a strategy about when and where to add or remove simulated vehicles (called *estimated vehicles*) in the microscopic simulation in order to make the actual behavior of the probe vehicles align with their expected behavior predicted by their car-following models. The results show that the effective penetration rate can be increased by around 20–30% using this method, which turned out beneficial for a ramp metering application. However, in this approach no principled method to deal with data or modeling errors is discussed.

1.2. Contribution and outline of this paper

The contribution of our paper is a generic data assimilation framework based on particle filters to reconstruct (plausible) vehicle trajectories on signalized urban arterials, that *does systematically address errors both in the measurements and in the model*. Like Goodall et al. (2016), our framework uses microscopic models of driving behavior and it assimilates *noisy* data from different sensors using particle filters (Arulampalam et al., 2002; Djurić et al., 2003). The framework does not impose restrictions on the type of microscopic models used; however, to illustrate the working, in this paper we consider the longitudinal movements of vehicles only. In terms of data, the method takes in noisy vehicle passages of individual vehicles (and as a result noisy vehicle counts) from loop detectors; signal timing parameters, and coarsely available travel time observations. By ‘noisy’, we mean that the passage data contains miss- and over-counts, resulting in counting errors. In this paper, an extensive simulation study is conducted to test the proposed data assimilation framework for trajectory reconstruction. The results show that the proposed method produces reasonable results for the

vehicle trajectories; it is also shown that the proposed method is robust to errors both in the model and in the measurements.

The rest of this paper is organized as follows. We first overview the main notation used throughout this paper. Section 2 then overviews the overall trajectory reconstruction methodology, after which Section 3 details the particle filter based data assimilation framework and Section 4 discusses data preprocessing methods and assumptions related to error models associated with the different data sources used. The simulation study to test the framework is presented in Section 5. In Section 6, sensitivity analysis (in terms of the sensor errors, the model errors and the number of particles) and possible extensions (such as vehicle accumulations and additional data) are discussed. Finally, the paper is concluded in Section 7.

1.3. Notation

Name	Unit	Description
ΔT	s	size of one discrete time step, i.e. the measurement interval
$N(t)$	veh	number of vehicles at time instant t
N_k	veh	number of vehicles at time step k, $N_k \equiv N(k\Delta T)$
$Q(t)$	veh	number of vehicles that have entered the simulation till time instant t
Q_k	veh	number of vehicles that have entered the simulation till time step k, $Q_k \equiv Q(k\Delta T)$
$x^i(t)$	m	location of the i-th vehicle at time instant t
x_k^i	m	location of the i-th vehicle at time step k, $x_k^i \equiv x^i(k\Delta T)$
$v^i(t)$	m/s	speed of the i-th vehicle at time instant t
v_k^i	m/s	speed of the i-th vehicle at time step k, $v_k^i \equiv v^i(k\Delta T)$
$s^i(t)$	–	state of the i-th vehicle at time instant t
s_k^i	–	state of the i-th vehicle at time step k, $s_k^i \equiv s^i(k\Delta T)$
S_k	–	system state at time step k
$E_{k,\Delta T}$	–	vehicle arrivals during the k-th measurement interval $[(k-1)\Delta T, k\Delta T]$
p	–	detection accuracy of a sensor
λ	s^{-1}	occurrence rate of over-count
m_k^o	–	measurement at time step k
N_s	–	number of sensors
$\mathscr{L}$	–	set of road stretches in the system
E_k^j	–	observed passages from the j-th ($1 \leqslant j \leqslant N_s$) sensor during $[(k-1)\Delta T, k\Delta T]$
N_k^l	veh	observed number of vehicles on road stretch l ($l \in \mathscr{L}$) at time step k
S_k^i	–	system state represented by the i-th particle at time step k
$E_{k,\Delta T}^i$	–	reconstructed vehicle arrivals for the i-th particle during the k-th measurement interval $[(k-1)\Delta T, k\Delta T]$
w_k^i	–	weight of the i-th particle at time step k
N_k^i	veh	number of vehicles in the i-th particle at time step k
N_p	–	number of particles
$x^{i,j}(t)$	m	location of the j-th vehicle in the i-th particle at time instant t
$x_k^{i,j}$	m	location of the j-th vehicle in the i-th particle at time step k, $x_k^{i,j} \equiv x^{i,j}(k\Delta T)$
$v^{i,j}(t)$	m/s	speed of the j-th vehicle in the i-th particle at time instant t
$v_k^{i,j}$	m/s	speed of the j-th vehicle in the i-th particle at time step k, $v_k^{i,j} \equiv v^{i,j}(k\Delta T)$
$p(s_{0:k}\|m_{1:k})$	–	posterior distribution
$p(s_k\|s_{k-1})$	–	system transition density
$p(m_k\|s_k)$	–	conditional probability of measurement m_k given state s_k
$q(s_{0:k}\|m_{1:k})$	–	importance density
$E_k^{i,j}$	–	vehicle passing times from the j-th ($1 \leqslant j \leqslant N_s$) sensor during the k-th measurement interval $[(k-1)\Delta T, k\Delta T]$ when generating the i-th particle
$N_k^{i,l}$	veh	number of vehicles on road stretch l ($l \in \mathscr{L}$) at time step k in the i-th particle
N_m	–	number (random variable) of miss-counts
N_o	–	number (random variable) of over-counts
$Q_i(t)$	veh	cumulative counts from the detector at cross-section x_i at time instant t
$t_i(n)$	s	time instant when the n-th vehicle passes cross-section x_i
$T_r(n)$	s	travel time estimated from the n-th vehicle
$T_r^{obs}(t)$	s	travel time observation observed at time instant t
$p(\{E_k^j\}_{j=1}^{N_s}\|S_{k-1:k}^i)$	–	conditional probability of vehicle passages given $S_{k-1:k}^i$
$p(\{N_k^l\}_{l=1}^{\mathscr{L}}\|S_{k-1:k}^i)$	–	conditional probability of vehicle accumulation given $S_{k-1:k}^i$

$\overline{E}^{j}_{number}$	%	average estimation error of the number of vehicle passages at sensor j
$\overline{E}^{j}_{match}$	%	average percentage of passages that are not accurately reconstructed at sensor j
$\overline{E}_{flow}$	%	average percent error of the traffic flow estimation
$\overline{E}_{density}$	%	average percent error of the traffic density estimation
$\overline{E}^{t}_{tr}$	%	average translation error of the reconstructed trajectories
$\overline{E}^{v}_{tr}$	%	average distortion error of the reconstructed trajectories
$\overline{E}^{t,v}_{tr}$	%	average overall error of the reconstructed trajectories
$P(X = x)$	–	discrete probability distribution of random variable X
$P(X\|Y)$	–	conditional probability
$\widehat{X}$	–	estimation of variable X

2. Overview: a generic data assimilation framework for trajectory reconstruction

In this section we describe the objective and overall ideas of the data assimilation framework. The objective of the framework is to estimate the most probable system state trajectory (in our case: the most probable set of vehicle trajectories on a roadstretch $[x_1,x_2]$ over a time period T), given the available data. The particle filter does this by estimating the posterior distribution with a sufficient set of support points (particles) spanning this distribution, each of which represents one possible set of vehicle trajectories that is computed by an arbitrary microscopic simulation model. One can interpret this posterior as a "belief histogram" of the system state trajectory, that describes a sample of probable sets of vehicle trajectories, given the data. This interpretation, however, is not very informative in the context of trajectory reconstruction. We do not want many probable sets of vehicle trajectories, but *one* set that is most likely given the data. Therefore we output the set of trajectories with the highest weight, although this is not strictly correct (since we obtain the estimation of interest not by Monte Carlo integration, but by the particle with the highest weight).

The main idea and ingredients of our approach are depicted in Fig. 1. Reconstructing vehicle trajectories with this approach thus implies attempting to reconstruct the *correct* number of *plausible* vehicle locations, speeds (and possibly other dynamic attributes). The *italics* indicate that this reconstruction involves two aspects:

1. We need to estimate the correct number of vehicles with the correct departure time on the basis of observed vehicle passages. The key problem to overcome here is that vehicle counts contain errors. Counting errors (caused by miss- or over-counting passages) have a large effect on reconstructing trajectories. One miss- or over-count may yield errors in *all* subsequent trajectories. Our approach to address this problem is to first make the number of vehicle passages (by extension the flows) between two detectors consistent. To this end we use the method proposed by van Lint and Hoogendoorn (2015), which—similarly to the method described in Bhaskar et al. (2014)— uses coarsely available individual travel times to periodically correct cumulative curves (at location x_1 and x_2 respectively) and as a consequence compute a correct number of vehicles between the detectors.
2. Using these corrected vehicle counts we can now reconstruct plausible vehicle trajectories—potentially aided by either

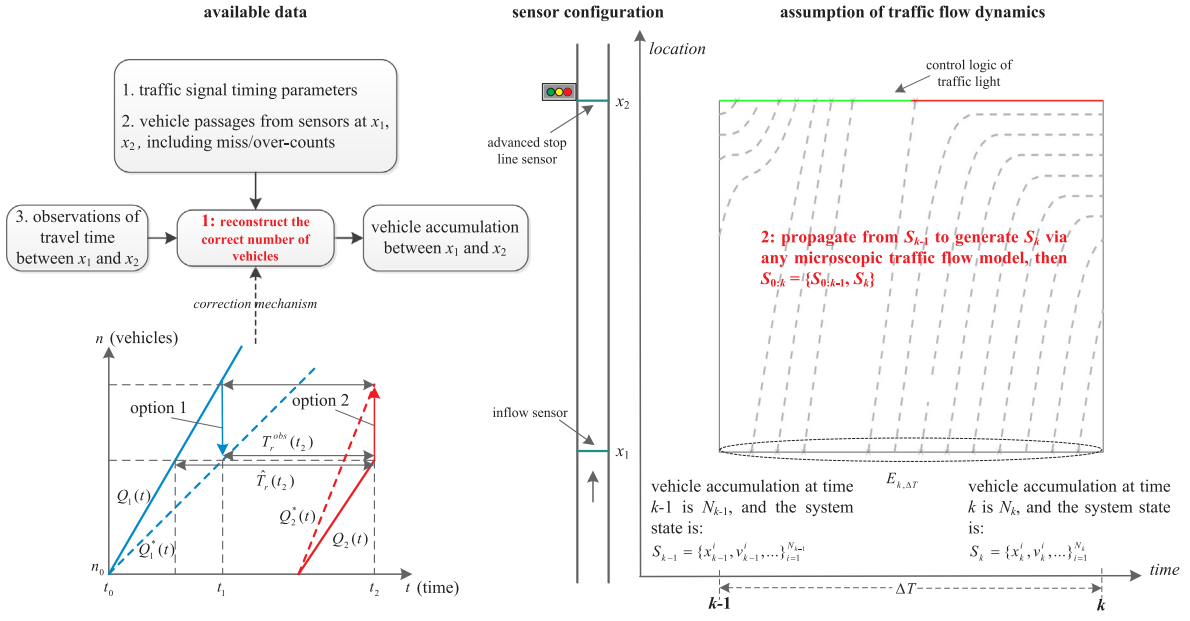


Fig. 1. The main idea of the data assimilation framework for vehicle trajectory reconstruction (ΔT is the measurement interval).

macroscopic speed data or individual trajectory data. As touched upon in the introduction, this can be done by using macroscopic (first or higher order) traffic flow theory, or by using microscopic traffic flow simulation models (or anything in between). We choose the latter. In fact, our framework allows the analyst to use whatever microscopic model deemed appropriate (for the application at hand); that is, a model with longitudinal behavior only, or a full fletched model that includes lane changing, crossing, etc. *Clearly, more degrees of freedom, without associated evidence in the form of microscopic data, may yield more sophisticated estimations but not necessarily more plausible vehicle trajectories.*

Since many of the underlying steps are interrelated, we discuss these aspects in a specific order. In Section 3 we first explain the particle filtering rationale (aspect 2). This involves the basic particle filtering principle, resampling and the procedure to go from partial vehicle trajectories (reconstructed whenever new measurements are available) to full trajectories (over consecutive measurement time intervals). Thereafter, in Section 4, we discuss the measurements (including the vehicle accumulation correction method), and the assumptions related to the errors in each used data source (aspect 1). We then conclude Section 4 with the computation of particle weights (part of aspect 2), that depends on these error models.

3. Particle filter design for trajectory reconstruction

3.1. Particle filter and resampling design

Consider the following discrete state dynamics

$$s_k = f_k(s_{k-1}) + \nu_{k-1}, k = 1,2,..., \tag{1}$$

in which f_k is a (possibly nonlinear) function of the state vector s_{k-1}, and ν_{k-1} represents a system noise process. At time k, measurements are available

$$m_k = g_k(s_k) + \varepsilon_k, k = 1,2,..., \tag{2}$$

in which g_k is a (possibly nonlinear) function that maps the state to the measurements, and ε_k represents a measurement noise process.

The objective of the particle filter is to estimate the conditional distribution of all states up to time k (i.e. the state trajectory) given all available measurements up to k, i.e. $p(s_{0:k}|m_{1:k})$, where $s_{0:k} = \{s_i, i = 0,1,2,...,k\}, m_{1:k} = \{m_j j = 1,2,...,k\}$. Since an analytic solution of $p(s_{0:k}|m_{1:k})$ is usually intractable, we generate a set of Monte Carlo samples (particles) with their associated weights to approximate this posterior distribution. If the number of particles is sufficiently large, the posterior can be approximated to an arbitrary accuracy (Arulampalam et al., 2002; Djurić et al., 2003; van Leeuwen, 2009). With this sample of particles all relevant statistical moments can be obtained using standard Monte Carlo integration techniques.

Let $\chi_k = \{s^i_{0:k}, w^i_k\}_{i=1}^{N_p}$ represent a random measure that characterizes the posterior distribution $p(s_{0:k}|m_{1:k})$, where $\{s^i_{0:k}\}_{i=1}^{N_p}$ is a set of support points (particles), and $\{w^i_k\}_{i=1}^{N_p}$ the set of associated weights. Then $p(s_{0:k}|m_{1:k})$ can be approximated as

$$p(s_{0:k}|m_{1:k}) \approx \sum_{i=1}^{N_p} w^i_k \delta(s_{0:k} - s^i_{0:k}) \tag{3}$$

where $\delta(\cdot)$ is the Dirac delta function. For our application, $s^i_{0:k}$ represents one possible set of vehicle trajectories over time–space window $[0,k\Delta T] \times [x_1,x_2]$ (see Fig. 1), and w^i_k is the weight of that set given all observations (vehicle passages, vehicle accumulations). If we can generate the particles $\{s^i_{0:k}\}_{i=1}^{N_p}$ from $p(s_{0:k}|m_{1:k})$, each of them will be assigned a weight equal to $1/N_p$. However, direct sampling from $p(s_{0:k}|m_{1:k})$ is usually intractable. An alternative is to generate the particles from a distribution $q(s_{0:k}|m_{1:k})$, known as *importance density* (Arulampalam et al., 2002; Djurić et al., 2003; van Leeuwen, 2009), and assign weights according to

$$w^i_k = \frac{p(s^i_{0:k}|m_{1:k})}{q(s^i_{0:k}|m_{1:k})}.$$

Based on Bayes' theorem, $p(s_{0:k}|m_{1:k})$ can be expressed as $p(s_{0:k}|m_{1:k}) = \frac{p(s_{0:k})p(m_{1:k} \mid s_{0:k})}{p(m_{1:k})}$. Similarly we have $p(s_{0:k-1}|m_{1:k-1}) = \frac{p(s_{0:k-1})p(m_{1:k-1} \mid s_{0:k-1})}{p(m_{1:k-1})}$. Therefore we can obtain a sequential update equation as

$$p(s_{0:k}|m_{1:k}) = \frac{p(m_k|s_k)p(s_k|s_{k-1})p(s_{0:k-1}|m_{1:k-1})}{p(m_k|m_{1:k-1})} \propto p(m_k|s_k)p(s_k|s_{k-1})p(s_{0:k-1}|m_{1:k-1}).$$

In case the importance density is chosen to factorize such that $q(s_{0:k}|m_{1:k}) = q(s_k|s_{0:k-1}, m_{1:k})q(s_{0:k-1}|m_{1:k-1})$, the random measure $\chi_{k-1} = \{s^i_{0:k-1}, w^i_{k-1}\}_{i=1}^{N_p}$ can be updated sequentially whenever new measurements m_k become available. The procedure then becomes:

- Obtain samples $s^i_{0:k} \sim q(s_{0:k}|m_{1:k})$ by augmenting samples from previous time step $s^i_{0:k-1} \sim q(s_{0:k-1}|m_{1:k-1})$ with the new state $s^i_k \sim q(s_k|s^i_{0:k-1}, m_{1:k})$;
- Update weights by

$$w^i_k = \frac{p(s^i_{0:k}|m_{1:k})}{q(s^i_{0:k}|m_{1:k})} \propto \frac{p(m_k|s^i_k)p(s^i_k|s^i_{k-1})p(s^i_{0:k-1}|m_{1:k-1})}{q(s^i_k|s^i_{0:k-1}, m_{1:k})q(s^i_{0:k-1}|m_{1:k-1})} = \frac{p(m_k|s^i_k)p(s^i_k|s^i_{k-1})}{q(s^i_k|s^i_{0:k-1}, m_{1:k})} w^i_{k-1}$$

If we assume that $q(s_k|s_{0:k-1},m_{1:k}) = q(s_k|s_{k-1},m_k)$, i.e., the importance density depends on s_{k-1} and m_k *only*, we have

$$w_k^i \propto \frac{p(m_k|s_k^i)p(s_k^i|s_{k-1}^i)}{q(s_k^i|s_{k-1}^i,m_k)} w_{k-1}^i \tag{4}$$

A pragmatic choice for the importance density is the system transition density, i.e. $q(s_k|s_{k-1},m_k) = p(s_k|s_{k-1})$—this is the approach we choose in this study. As a result, Eq. (4) simplifies to

$$w_k^i \propto p(m_k|s_k^i) w_{k-1}^i \tag{5}$$

A major problem of particle filters is that the discrete random measure degenerates quickly (Arulampalam et al., 2002; Djurić et al., 2003; van Leeuwen, 2009). In other words, all the particles except for a very few are assigned negligible weights. The solution is to resample the particles after they are updated. There exist different resampling algorithms and methods to determine when resampling is necessary (Arulampalam et al., 2002). A simple and often adopted resampling method is to replicate particles in proportion to their weights. It has been shown that a sufficient large number of particles are able to converge to the true posterior distribution even in nonlinear, non-Gaussian dynamic systems (Arulampalam et al., 2002; Djurić et al., 2003; van Leeuwen, 2009). We propose a hierarchical strategy that is tailored to our specific problem.

Recall that the overall objective of our assimilation framework (Fig. 1) is to estimate the most probable number of vehicles; and—given this number of vehicles—estimate the most probable vehicle trajectories. The rationale of our resampling strategy is based on these two aspects. We first categorize particles into groups with similar vehicle accumulations, and then resample within each group. This will ensure there is a sufficient number of particles selected over a range of possible vehicle accumulations. It is important to note that the weights of particles within such a group do not differ as much compared to when all particles are considered as a whole. This implies that by sampling within each group, the particles with small weights will have a higher probability to be selected than in case all particles are resampled together. As a result, this hierarchical resampling method will ensure that many likely particles are kept for future iterations, whereas a sufficient number of less likely particles are not discarded. The consequence is a better coverage of the entire state space. Suppose that the particles generated from the sampling step are $\{S_{0:k}^i, w_k^i\}_{i=1}^{N_p}$, the hierarchical resampling method now works as follows:

- Sample based on the vehicle accumulations (number of vehicles) at time k. Suppose that at time k, the vehicle accumulation in the i-th particle is N_k^i. We categorize particles into groups based on their vehicle accumulation value $N_k^i, i = 1,2,...,N_p$, such that any two particles within the same group have the same value of vehicle accumulation. Assume that there are in total $C (C \leqslant N_p)$ such groups, and the particles in the c-th group have equal value of vehicle accumulation, which is denoted as $N^c, c = 1,2,...,C$. Then we determine how many particles should be selected from the c-th group. This is done in the following way:
  - ◇ We first assign a probability value $p_c = P(E = N^c - N_k)$ to the c-th group, where N_k is the real vehicle accumulation at time k, and $P(E)$ corresponds to the error model for vehicle accumulation, which will be discussed in Section 4.2. Then we normalize these probability values such that $\sum_{c=1}^{C} p_c = 1$. Consequently we obtain a discrete probability distribution $P(X = c) = p_c, c = 1,2,...,C$.
  - ◇ We then sample the discrete probability distribution $P(X)$ N_p times, and obtain samples $\{c_1,c_2,...,c_{N_p}\}$. The j-th sample c_j indicates that when selecting particles, the j-th selection should select a particle from the c_j-th group, where $j = 1,2,...,N_p; c_j \in \{1,2,...,C\}$. Given samples $\{c_1,c_2,...,c_{N_p}\}$, we can easily know the number of particles that should be selected from the c-th group, and we denote this number as $Nr(c), c = 1,2,...,C$. Then we have $\sum_{c=1}^{C} Nr(c) = N_p$.
- Sample based on the particle weights within each group. In the c-th group, we draw $Nr(c)$ particles with probabilities proportional to their weights (the weights are temporarily normalized within the group for sampling). In this process, if a particle is selected, its original weight is also associated with that particle. When all groups are sampled, the particles selected from each group form the resampled particles $\{S_{0:k}^i\}_{i=1}^{N_p}$, and their associated weights are $\{w_k^i\}_{i=1}^{N_p}$.

In this section we elaborated on the sampling and weight updating principles; to actually compute a quantitative value for each weight, we require a mechanism that takes into account the errors associated with each observation. We therefore return to weight computation in Section 4.4 after discussing the data error models. In this section we further focus on the trajectory reconstruction method.

3.2. State dynamics: individual vehicle movements

In this section we specify the generic state dynamics in Eq. (1). In principle we can choose whatever traffic flow model deemed appropriate for the estimation task at hand, as long as it (endogenously) computes the system state from which measurements (Eq. (2)) can be constructed. Without loss of generality, in this paper, we consider the longitudinal movements of vehicles only, that is, we consider models for free-driving and car-following (CF) that describe the acceleration behavior of driver-vehicle units as a function of stimuli such as headway and speed difference with respect to a leading vehicle, subject to physical considerations, and a wide range of (behavioral) assumptions. In general terms, (discretized) CF models take on the following form

$$a(t + T^r) = f(\eta(t),\zeta(t)),$$

in which $T^r \geqslant 0$ represents a reaction time, $\eta(t)$ is a set of stimuli (headway, speed difference with leader, etc.), and $\zeta(t)$ is a set of parameters specifying above mentioned behavioral and physical assumptions. For comprehensive reviews of longitudinal models for driving behavior see e.g., van Wageningen-Kessels et al. (2015), Saifuzzaman and Zheng (2014), Brackstone and McDonald (1999). In our case, the precise form of the CF models (or the models for lane changing, crossing, etc.) does not matter. Clearly, applying a microscopic simulation model for state estimation, requires such a model to be well-calibrated. We refer the reader to Punzo and Montanino (2016), da Rocha et al. (2015), Punzo et al. (2015) for an in-depth treatment of microscopic model calibration, which is beyond the scope of this paper. For our purposes, it is sufficient to describe the resulting vehicle trajectory by time series of two variables: longitudinal location $x^i(t)$ and speed $v^i(t)$, where i is index of the vehicle, that is

$$s^i(t) = \{x^i(t), v^i(t)\}, t \in \mathbb{R}^+_{0,\infty} = \{r \in \mathbb{R} | r \geqslant 0\}. \tag{6}$$

During the simulation, new vehicles continuously arrive at the system boundary and enter the simulation. These arrivals during a ΔT length period $[(k-1)\Delta T, k\Delta T], k = 1,2,...$, are modeled as a list of arrival events, that is,

$$E_{k,\Delta T} = \{e_1, e_2, ..., e_n\}, \tag{7}$$

where $e_i = \{t_{e_i}, A_{e_i}\} (i = 1,2,...,n)$ represents an arrival event of a new vehicle, where t_{e_i} is the relative arrival time to the beginning of the period (i.e. $(k-1)\Delta T$), and A_{e_i} is a set of attributes of the new vehicle, such as speed, destination, route, and vehicle type. Note that in principle, such additional attributes could be part of the state vector, given observations are available to estimate these. This is beyond the scope of this paper.

Now, we can define the simulation state at time k as

$$S_k = \{s_k^i\}_{i=1}^{Q_k}, k = 0,1,2,..., \tag{8}$$

where $s_k^i \equiv s^i(k\Delta T)$ (see Eq. (6)), and $Q_k = N_0 + \sum_{j \in Inflow} \int_0^{k\Delta T} q_j(s) ds$ is the number of vehicles that have entered the simulation till time k. N_0 is the number of vehicles in the simulation initialization; *Inflow* is a set of system boundaries that vehicles can flow in, and $q_j(s)$ is the corresponding flow. Notice that S_k still includes the states of vehicles that have left the system. However, once a vehicle leaves the system (i.e. the simulation study area), we stop updating its state in the simulation. The evolution of the total state of the simulation (and thereby Eq. (1)) can now be expressed as

$$S_k = SIM(S_{k-1}, E_{k,\Delta T}) + \nu_{k-1}, k = 1,2,..., \tag{9}$$

where *SIM* is the microscopic urban traffic simulation, that contains models for driving behavior and traffic control logic, and $S_k, E_{k,\Delta T}$ are defined in Eqs. (8) and (7), respectively.

3.3. Specification of the framework for trajectory reconstruction

Now consider an urban traffic network with N_s sensors deployed at each inflow boundary of a lane and each stop line at signalized junctions. The objective is to determine time series of every vehicle's location and speed over all links using sensor data. Below we first discuss the data we need for our method; then the procedure to estimate partial trajectories during measurement periods of size ΔT; and finally the procedure to concatenate vehicle trajectories over multiple measurement periods.

3.3.1. Available data

This data assimilation framework assumes the availability of minimally the following data:

- Vehicle passages. As shown in Fig. 1, sensors are placed at each inflow boundary of a lane and each advanced stop line at signalized junctions. We assume that the data provided by sensors is event-based data, i.e., a sequence of individual vehicle's passage times. We assume these event-based data contain errors such as miss-counts (a vehicle passes by, but the sensor fails to detect its passage) and over-counts (no vehicles pass by, but the sensor reports a passage).
- Traffic signal timing parameters. These data are input when running the traffic simulation model.
- Periodic travel time observations. These data are used in the correction method to estimate vehicle accumulation.

We assume data are available every ΔT time units. The measurements at time k are denoted by

$$m_k^o = \{\{E_k^j\}_{j=1}^{N_s}, \{N_k^l\}_{l=1}^{\mathscr{L}}\}, k = 1,2,..., \tag{10}$$

where N_s is the number of sensors, and E_k^j depicts the passage times obtained from sensor j during time interval $[(k-1)\Delta T, k\Delta T]$. Note that vehicle accumulation is not measured directly but estimated from vehicle counts, which will be explained in Section 4.1. The symbol $\mathscr{L}$ denotes a set of road stretches for which such vehicle accumulation is available, and N_k^l depicts the vehicle accumulation on road stretch l at time k. Traffic signal timing data and travel time observations are not used in weight computation, therefore we omit them in Eq. (10).

3.3.2. Partial trajectory reconstruction

Recall that particle filtering essentially approximates the posterior distribution $p(S_{0:k}|m_{1:k}^o)$, in which S_k is defined in Eq. (8). However, Q_k (see Eq. (8)) is random, and therefore the dimension $S_{0:k}$ is also random. This randomness results in the so-called

variable dimension problem (Godsill et al., 2007), which *in principle* makes the sequential importance sampling algorithm introduced in Section 3.1 not applicable to our problem. However, it turns out that, in line with Godsill et al. (2007), we can formally prove that the variable dimension sequence has no tangible effect on weight updating in our case, and we therefore can safely apply the sequential importance sampling algorithm. Due to space limitations, the proof is put in Appendix A.

Since the measurements (Eq. (10)) are not directly related to a system state at one specific time instant, but are related with a sequence of system states over a period of time, the measurement model is formalized as

$$m_k^o = g_k(S_{k-1:k}) + \varepsilon_k, k = 1,2,..., \tag{11}$$

where $S_{k-1:k}$ represents a sequence of (intermediate) system states from time $k-1$ to time k (sampled every $\delta t \ll \Delta T$ time units during $[(k-1)\Delta T, k\Delta T]$). Note that $S_{k-1:k}$ essentially defines a set of (sampled) vehicle trajectories during $[(k-1)\Delta T, k\Delta T]$. Accordingly, this also implies that the particle weights should be updated as

$$w_k \propto p(m_k^o | S_{k-1:k}) w_{k-1}.$$

Note that although the state evolution after time k depends on S_k only, the application at hand requires us to store the system state trajectory for as many time periods as needed to reconstruct trajectories. Algorithm 1 describes in detail how a particle filter is applied to fulfill the trajectory reconstruction task. The main steps of the proposed algorithm are summarized as below.

- **Initialization**. In the initialization step (line 2–5 in Algorithm 1), the i-th sample S_0^i is actually a guess of vehicles' locations and speeds in the network, i.e., $S_0^i = \{x_0^{i,j}, v_0^{i,j}\}_{j=1}^{N_0^i}$, where N_0^i is the vehicle accumulation (i.e. the number of vehicles) in the i-th sample at time $k = 0$.
- **Sampling**. After initialization, the microscopic traffic simulation model is run for one time step ΔT (i.e. until new measurements become available) to obtain a sample (line 8 in Algorithm 1). This is done for every particle. To run the simulation, inputs (vehicle arrivals) are reconstructed from the (noisy) passage times observed by sensors deployed at the inflow boundary. During the run, intermediate states $S_{k-1:k}^i$ are recorded, which represent a set of (sampled) vehicle trajectories that will be used in weight computation and subsequent trajectory reconstruction. Once a sample is generated, its weight is updated based on the newly available measurements (line 9 in Algorithm 1).
- **Resampling**. To solve the degeneracy problem, we resample the particles on a regular basis (see Section 3.1).
- **Estimation**. We reconstruct partial trajectories from the particle with the highest weight. Note that we elaborate in detail in Section 3.3.3 how these partial trajectories are subsequently concatenated to full trajectories.

Algorithm 1. The particle filter for vehicle trajectory reconstruction.

1 % initialization of particles at $k = 0$
2 **for** $i = 1 : N_p$ **do**
3 | generate the i-th sample S_0^i
4 | set weight $w_0^i = 1/N_p$
5 **end**
6 % the sampling step for any time $k \geq 1$
7 **for** $i = 1 : N_p$ **do**
8 | prediction step: reconstruct a sequence of vehicle arrivals $E_{k,\Delta T}^i$, and run the simulation for one time step (ΔT) with initial state S_{k-1}^i, then S_k^i is generated. The i-th particle $S_{0:k-1}^i$ is thus expanded to $S_{0:k}^i = \{S_{0:k-1}^i, S_k^i\}$; meanwhile, the intermediate states $S_{k-1:k}^i$ are also recorded for weight computation and trajectory reconstruction
9 | update step: assign $S_{0:k}^i$ a weight: $w_k^i = p(m_k^o | S_{k-1:k}^i) w_{k-1}^i$
10 **end**
11 normalize the weights, denote them as $\{S_{0:k}^i, w_k^i\}_{i=1}^{N_p}$
12 % the resampling step (more details are explained in section 3.1)
13 $\{S_{0:k}^i, w_k^i\}_{i=1}^{N_p} = hierarchical\_resampling(\{S_{0:k}^i, w_k^i\}_{i=1}^{N_p})$
14 % output latest trajectories
15 find the particle with the maximum weight: $w_k^{i^*} = max\{w_k^i | i = 1, \ldots, N_p\}$
16 output trajectories recorded in the simulation which generated the i^*-th particle, and update previous reconstruction results; more details are presented in section 3.3.3
17 **for** $i = 1 : N_p$ **do**
18 | $w_k^i = 1/N_p$
19 **end**

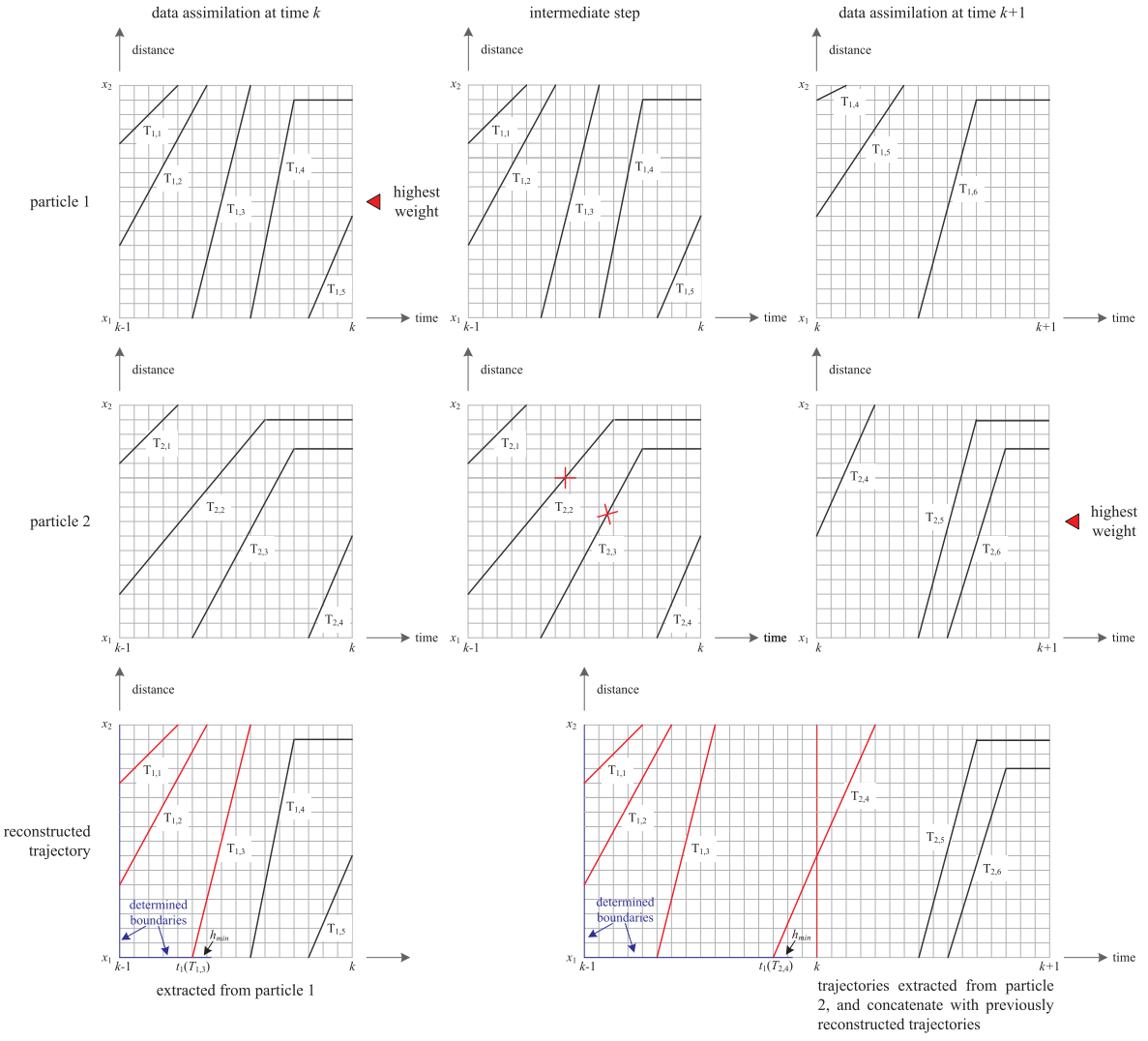


Fig. 2. Illustration of full vehicle trajectory reconstruction (the duration between two consecutive time steps is ΔT).

3.3.3. *Full trajectory reconstruction*

The final step is now to concatenate partial trajectories over multiple measurement intervals into full trajectories over the desired time–space window. This full trajectory reconstruction procedure is graphically illustrated in Fig. 2, in which for illustration purposes, only two particles are considered. The evolutions of particle 1 and particle 2 are depicted respectively at the top and in the middle of Fig. 2, whereas the reconstructed vehicle trajectories are placed at the bottom part.

At time k, we assume that the first particle has the highest weight (see line 15 in Algorithm 1). Therefore, $\{T_{1,1},T_{1,2},T_{1,3},T_{1,4},T_{1,5}\}$ from particle 1 (these trajectories are recorded when generating the particle; see line 8 in Algorithm 1) now constitute the reconstructed trajectories (see line 16 in Algorithm 1) in time–space region $[(k-1)\Delta T,k\Delta T]\times[x_1,x_2]$, as shown in the bottom-left plot in Fig. 2. Among these trajectories, $T_{1,1},T_{1,2}$, and $T_{1,3}$ have passed cross-section x_2, and therefore, they are fully evaluated given all available evidence. This implies that information estimated by particle 1 (the particle with the highest weight) at boundaries

$$\begin{aligned}&\{distance\in[x_1,x_2]\cap time=(k-1)\Delta T\}\\&\{distance=x_1\cap time\in[(k-1)\Delta T,t_1(T_{1,3})+h_{min}]\}\end{aligned}\tag{12}$$

is already the best estimation of the true boundary information, with $t_1(T_{1,3})$ denoting the time instant when the last full trajectory $T_{1,3}$ crosses x_1, and h_{min} denoting the minimum time headway (we add h_{min} to $t_1(T_{1,3})$ to prevent collisions). We call these boundaries *determined boundaries*, as shown in Fig. 2. We temporarily keep $T_{1,4}$ and $T_{1,5}$ in our reconstructed results. In case data assimilation stops at time k, the trajectory set $\{T_{1,1},T_{1,2},T_{1,3},T_{1,4},T_{1,5}\}$ is the best estimation based on the observed evidence; whereas in case new data (passages) are available in the next iteration, information at boundary $\{distance\in[x_1,x_2]\cap time=k\Delta T\}$ (originating from the undetermined boundary $\{distance=x_1\cap time\in[t_1(T_{1,3})+h_{min},k\Delta T]\}$) will be re-evaluated.

Since information at the determined boundaries defined in Eq. (12) has been determined, we introduce an intermediate step as

shown in the middle column in Fig. 2. In this step, we check in each particle (except the particle with the highest weight that has defined the determined boundary) whether there is a trajectory originating from the determined boundaries, i.e., to check if the trajectory meets the following two conditions:

- does it cross with boundary $\{distance \in [x_1,x_2] \cap time = k\Delta T\}$?
- does it originate from boundaries defined in Eq. (12)?

If any trajectory meets these conditions, we delete it. As shown in Fig. 2, we delete $T_{2,2}$ and $T_{2,3}$ from particle 2. This step ensures that in future iterations, newly reconstructed trajectories will not influence trajectories which were reconstructed in past iterations.

At the next time step, i.e. $k + 1$, we assume that the second particle has the highest weight. Therefore, we now populate time–space region $[k\Delta T,(k + 1)\Delta T] \times [x_1,x_2]$ with trajectories extracted from particle 2, i.e., $\{T_{2,4},T_{2,5},T_{2,6}\}$. Note that we trace back the trajectories at boundary $\{distance \in [x_1,x_2] \cap time = k\Delta T\}$ to boundary $\{distance = x_1 \cap time = [t_1(T_{1,3}) + h_{min},k\Delta T]\}$; update the reconstruction results in the last iteration by deleting $T_{1,4}$ and $T_{1,5}$, and then concatenate the updated trajectories with the newly reconstructed trajectories (in this case $T_{2,4}$), as shown in the plot at the bottom-right of Fig. 2. In this case one full trajectory (passing x_2) has been reconstructed, and the determined boundaries are also expanded (see the bottom part in Fig. 2), and we wait for new evidence in the next iteration.

3.4. Computational complexity

The computational complexity of the particle filter based trajectory reconstruction method can be defined as $\mathcal{O}(N_p,S_{network},\overline{N}_{veh})$, where N_p represents the number of particles, $S_{network}$ defines the scale of the urban network, which can be quantified by the area in the urban network where vehicles can move, and $\overline{N}_{veh}$ indicates the average number of vehicles in the urban network. The computational complexity grows linearly with the number of particles, but exponentially with the scale of the urban network and the average number of vehicles in the network.

4. Vehicle count correction method, specification of error models and weight computation

In this section, we discuss the method to estimate vehicle accumulation and we specify our assumptions related to the error models for the various data sources. On the basis thereof, we then return to the computation of the particle weights.

4.1. Estimating vehicle accumulation

Consider two detectors installed at x_1 (upstream) and x_2 (downstream) on a closed roadstretch (no entry or exits), both measuring flow $q_i(t),i = 1,2$. Vehicle accumulation between x_1 and x_2 at any time instant t is equal to

$$N(t) = Q_1(t)-Q_2(t), \tag{13}$$

where $Q_i(t) = \int_0^t q_i(s)ds$ is the cumulative curve of the detector installed at $x_i,i = 1,2$. The travel time from x_1 to x_2 of the n^{th} vehicle equals $T_r(n) = t_2(n)-t_1(n)$, where $t_i(n) = Q_i^{-1}(n),i = 1,2$. In case of FIFO (first in, first out), T_r is exact, otherwise an approximation; $N(t)$ is exact in both cases. Unfortunately, if the detectors miss or over-count vehicle passages, that is $q_i^{true}(t) = q_i^{detector}(t) + \beta_{q_i}(t)$, with $\beta_{q_i}(t)$ an arbitrary noise term, both $N(t)$ and T_r will contain errors. Since $N(t)$ integrates consecutive flow counts (Eq. (13)) this is an additive error that—regardless of the precise form of $\beta_{q_i}(t)$—yields an unbounded estimation bias for $N(t)$. In van Lint and Hoogendoorn (2015) a method is proposed which uses travel time observations to correct the flows and thereby the cumulative counts to estimate the vehicle accumulation. This method works along the same lines as the CUPRITE method proposed in Bhaskar et al. (2014). The basic idea is shown in Fig. 3. Suppose the n_2^{th} vehicle passes x_2 at time t_2, the estimated travel time given by the cumulative curves is

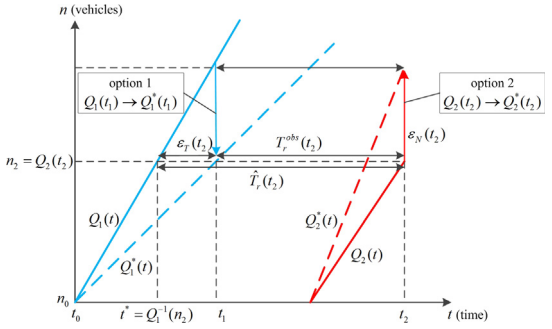


Fig. 3. Vehicle accumulation estimation using the correction mechanism.

$$\widehat{T}_r(t_2) = T_r(n_2) = t_2 - t^* = t_2 - Q_1^{-1}(n_2)$$

Assume at t_2, we observe a travel time $T_r^{obs}(t_2)$. The travel time error will be

$$\varepsilon_T(t_2) = T_r^{obs}(t_2) - \widehat{T}_r(t_2)$$

We can now use this travel time error to correct the error in the flows (and thus cumulative curves). We have two equivalent options (one error and two degrees of freedom), as shown in Fig. 3. The first is to rotate the cumulative curve $Q_1(t)$ such that $Q_1(t) \rightarrow Q_1^*(t)$ and the second is to rotate the cumulative curve $Q_2(t)$ such that $Q_2(t) \rightarrow Q_2^*(t)$, in both cases by an amount $\varepsilon_N(t_2)$. Let us assume we correct $Q_1(t)$ and that the previous correction took place for cumulative count $Q_1(t_0) = n_0$. Some straightforward algebra leads to

$$Q_1^*(t) = Q_1(t) + \frac{t - t_0}{t_1 - t_0}\varepsilon_N(t_2) \tag{14}$$

with

$$\varepsilon_N(t_2) = \frac{n_2 - n_0}{t^* - t_0}\varepsilon_T(t_2)$$

substituting (14) in (13) then gives

$$\widehat{N}(t) = Q_1^*(t) - Q_2(t)$$

This method yields very good results in terms of estimating vehicle accumulation, even when limited travel time measurements are available. Note that since $q(t) = \frac{dQ}{dt}$, this correction implies that all flow measurements $q(t)$, with $t \in [t_0, t_1]$ are effectively adjusted by this correction. However, since we have two degrees of freedom (up- and downstream flows) to adjust for one error, these adjusted flows may not be more accurate than before the correction. To generate vehicles on the roadstretch we therefore use the estimated vehicle accumulation $\widehat{N}(t)$ *only*, and *not* these corrected flows. Further below we detail how vehicles are generated and how the corrected cumulative counts of vehicles is used in our framework.

4.2. Error models for the different data sources

4.2.1. Vehicle accumulation

Although the method proposed above is able to remove the bias due to the additive errors in consecutive flow measurements, the resulting vehicle accumulations will still contain errors, which we can now assume to be zero mean (van Lint and Hoogendoorn, 2015). For the simulation case later on we approximate the discrete probability distribution of this estimation error by the histogram:

$$P(E = e) = \frac{Nr(e(t) = e)}{N}$$

in which $e(t)$ is the vehicle accumulation estimation error defined as

$$e(t) = N(t) - \widehat{N}(t), t = t_0, t_1, \ldots, t_{N-1}$$

where $N(t)$ is the ground truth value (obtained in the simulation) of the vehicle accumulation at time instant t, and $\widehat{N}(t)$ is its corresponding estimation. $Nr(e(t) = e)$ is the number of time instants when $e(t) = e$, and N is the number of time instants when the vehicle accumulation is estimated. Clearly, when applied in practice we need to assume a parameterized (discrete) distribution, for example fitted to this histogram.

4.2.2. Vehicle passages

There are two types of errors in vehicle passages, i.e., miss-counts and over-counts, and we model these two types of errors using two parameters:

- *detection accuracy* $p \in [0,1]$, depicting the probability that a sensor successfully detects a vehicle passage. Conversely, the probability that the sensor misses the passage equals $1-p$.
- *occurrence rate of over-count* λ. The number of over-counts during a time interval can be regarded as a Poisson distribution, and we define its occurrence rate as λ. As a result, the time between two consecutive over-count events is an Exponential distribution, with mean $1/\lambda$.

4.3. Arrival reconstruction

Assume that at time k, the available (noisy!) passage times from an inflow sensor are $E_k^j = \{t_1, t_2, \ldots, t_m\}$, where j is the index of the inflow sensor (see Eq. (10)). The problem we are going to solve in this section is to reconstruct a possible passage sequence based on the observation and the error model. We first apply a few simple rules to 'clean' the data from obvious errors. For example, if the time headway between two consecutive passage times on a single lane is (much) smaller than the minimum time headway, one of them is likely an over-count. A passage observed at the stop-line of an intersection during the red phase is over likely an over-count. In both

cases, we can clean the data by deleting these unlikely passage times. After this preprocessing step, we reconstruct a possible passage sequence E_k^j that is probable under the assumed error models for miss- and over-counts.

To this end, we first define a two-dimensional random variable: $X = (N_m, N_o)$, where N_m represents the number of miss-counts, and N_o the number of over-counts. We compute the joint probability distribution $p(X)$ using Algorithm 2. Once $p(X)$ is available, we draw (n_m, n_o) from $p(X)$ to reconstruct a possible passage sequence. Then we reconstruct passage times as follows:

- randomly delete n_o elements in E_k^j;
- randomly generate n_m time instants between $[(k-1)\Delta T, k\Delta T]$, until the time headways meet some predefined requirements (i.e., resultant time headway is larger than the minimum time headway); insert them into E_k^j.

Algorithm 2. Compute the joint probability distribution of miss-count and over-count.

Input: noisy passage times: E_k^j; sensor error model parameters: p, λ
Output: joint probability distribution: $p(X)$

% determine the probability distribution of the number of over-count $p(N_o)$
for $n_o \in \mathbb{N} \wedge n_o \leq |E_k^j|$ **do**
  compute $p_{n_o} = \frac{(\lambda\Delta T)^{n_o} e^{-\lambda\Delta T}}{n_o!}$
  if $p_{n_o} \geq 10^{-3}$ **then**
    $P(N_o = n_o) = p_{n_o}$
  end
end
normalize $p(N_o)$ to make it a probability distribution
for *every possible* n_o **do**
  determine the maximum number of miss-counts: $M = floor(\Delta T / h_m) - |E_k^j| + n_o$, where ΔT is the duration of the period during which E_k^j was collected, h_m is the minimum time headway, and $floor(x)$ returns the greatest integer less than or equal to x
  for $n_m \in \mathbb{N} \wedge n_m \leq M$ **do**
    compute $p_{n_m} = C_{|E_k^j|-n_o+n_m}^{|E_k^j|-n_o} p^{|E_k^j|-n_o}(1-p)^{n_m}$
    if $p_{n_m} \geq 10^{-3}$ **then**
      $P(N_m = n_m) = p_{n_m}$
    end
  end
  normalize $p(N_m)$ to make it a probability distribution
  for *every possible* n_m **do**
    $P(X = (n_m, n_o)) = p_{n_m} \times p_{n_o}$
  end
end
normalize $p(X)$ to make it a probability distribution

4.4. Weight computation: utilizing the error models

We can now return to the computation of the particle weights, which depends on the error models we introduced above. Recall that when a sample is generated, its weight is updated based on newly available measurement $m_k^o = \{\{E_k^j\}_{j=1}^{N_s}, \{N_k^l\}_{l=1}^{\mathscr{L}}\}$ (see detailed definition in Section 3.3.1). Since the two types of data are conditionally independent given $S_{k-1:k}^i$ (the intermediate states recorded when generating the i-th particle; see detailed explanation in Section 3.3.2), we have $p(m_k^o|S_{k-1:k}^i) = p(\{E_k^j\}_{j=1}^{N_s}|S_{k-1:k}^i)p(\{N_k^l\}_{l=1}^{\mathscr{L}}|S_{k-1:k}^i)$. Since measurements from different sensors are conditionally independent given $S_{k-1:k}^i$, we have $p(\{E_k^j\}_{j=1}^{N_s}|S_{k-1:k}^i) = \prod_{j=1}^{N_s} p(E_k^j|S_{k-1:k}^i)$. Similarly we have $p(\{N_k^l\}_{l=1}^{\mathscr{L}}|S_{k-1:k}^i) = \prod_{l\in\mathscr{L}} p(N_k^l|S_{k-1:k}^i)$, since vehicle accumulations on different road segments are conditionally independent given $S_{k-1:k}^i$. Consequently, the weight of the i-th particle is computed by

$$w_k^i = p(m_k^o|S_{k-1:k}^i)w_{k-1}^i = \underbrace{\prod_{j=1}^{N_s} p(E_k^j|S_{k-1:k}^i)}_{\text{vehicle passages}} \times \underbrace{\prod_{l\in\mathscr{L}} p(N_k^l|S_{k-1:k}^i)}_{\text{vehicle accumulation}} \times w_{k-1}^i,$$

which boils down to the computation of $p(E_k^j|S_{k-1:k}^i)$ and $p(N_k^l|S_{k-1:k}^i)$. Based on $S_{k-1:k}^i$, we can easily obtain (e.g., by linear

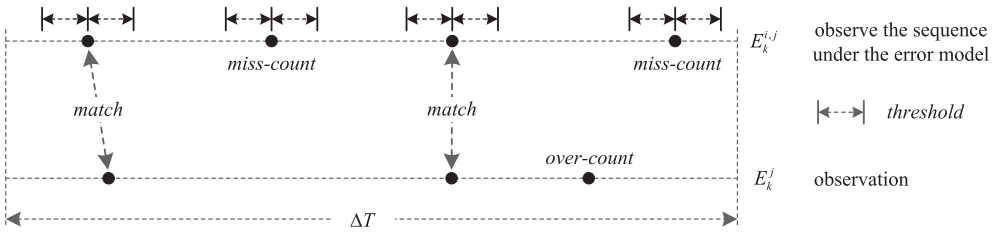


Fig. 4. Determine the miss-counts and over-counts.

interpolation) the estimated vehicle passages at sensor j, which is denoted as $E_k^{i,j}$; similarly we can quickly get (e.g., by counting) the estimated vehicle accumulation on road stretch l, which is denoted by $N_k^{i,l}$. Consequently, the computation of $p(E_k^j|S_{k-1:k}^i)$ and $p(N_k^l|S_{k-1:k}^i)$ boils down to the computation of $p(E_k^j|E_k^{i,j})$ and $p(N_k^l|N_k^{i,l})$, respectively, i.e., $p(E_k^j|S_{k-1:k}^i) = p(E_k^j|E_k^{i,j})$,$p(N_k^l|S_{k-1:k}^i) = p(N_k^l|N_k^{i,l})$. Therefore, in the rest of this section, we show how to compute $p(E_k^j|E_k^{i,j})$ and $p(N_k^l|N_k^{i,l})$ based on the error assumptions.

Assume that at time k, the available observation from the j-th sensor is $E_k^j = \{t_1,t_2,...,t_m\}$; at the same time, the estimated value from the i-th particle is $E_k^{i,j} = \{t'_1,t'_2,...,t'_{m_i}\}$. To compute $p(E_k^j|E_k^{i,j})$, we need to figure out which passages in $E_k^{i,j}$ are miss-counted, and which passages in E_k^j are over-counts. To this end, a match procedure is applied, which is essentially the same with that in Wang and Hu (2015), where the authors use it to match two groups of agents (without identity information) based on their locations. The match procedure works in the following way. Given set $X = \{x_1,x_2,...,x_m\}$, set $Y = \{y_1,y_2,...,y_n\}$, and a threshold value *threshold*, we first calculate the distances between all possible pairs $\langle x_i,y_j\rangle$,$\forall\ x_i \in X,y_j \in Y$. We then sort the pairs $\langle x_i,y_j\rangle$ in ascending order based on the distance (defined by a distance function $d(\cdot,\cdot)$) between x_i and y_j, and store the sorted pairs in a list L. After that, we recursively select the first element $\langle x_{i_1},y_{j_1}\rangle$ in L (i.e. the pair with the smallest distance), and if the distance $d(x_{i_1},y_{j_1}) < threshold$, we store $\langle x_{i_1},y_{j_1}\rangle$ in a set MS (referred to as the *matched set*), and then delete all the pairs in L that contains x_{i_1} or y_{j_1}. This continues until L is empty or the first pair in L has a distance larger than or equal to the predefined threshold value *threshold*. We formally denote this match procedure as $MS = match(X,Y,threshold)$.

We apply the match procedure on $E_k^{i,j}$ and E_k^j, with the distance function defined as $d(t,t') = |t-t'|$, where $t \in E_k^j$,$t' \in E_k^{i,j}$, and the threshold value chosen as the minimum time headway h_{min}. After the match, passages in $E_k^{i,j}$ that are not matched will be regarded as miss-counts, while passages in E_k^j that are not matched will be regarded as over-counts, which are shown in Fig. 4. Then $p(E_k^j|E_k^{i,j})$ is calculated as follows:

$$p(E_k^j|E_k^{i,j}) = \underbrace{e^{-d_m}}_{\text{penalty}} \times \underbrace{p^{m_i-n_m} \times (1-p)^{n_m}}_{\text{miss-count}} \times \underbrace{\frac{(\lambda\Delta T)^{n_o}e^{-\lambda\Delta T}}{n_o!}}_{\text{over-count}},$$

where $d_m = max\{d(t,t')|\langle t,t'\rangle \in MS = match(E_k^j,E_k^{i,j},h_{min})\}$,$m_i$ is the number of passages in $E_k^{i,j}$,n_m is the number of miss-counts, and n_o is the number of over-counts. Among these numbers, the relation $|MS| = n-n_o = m_i-n_m$ holds, where n is the number of passages in E_k^j. The term e^{-d_m} acts as a penalty function, which means that we favor those particles that have smaller distance to the observation. The term $p^{m_i-n_m} \times (1-p)^{n_m}$ computes the probability of miss-counts, while the term $\frac{(\lambda\Delta T)^{n_o}e^{-\lambda\Delta T}}{n_o!}$ computes the probability of over-counts.

The computation of $p(N_k^l|N_k^{i,l})$ finally yields

$$p(N_k^l|N_k^{i,l}) = P(E^l = N_k^{i,l}-N_k^l),$$

where N_k^l is the vehicle accumulation on road segment l at time k estimated by the correction method, while $N_k^{i,l}$ is the corresponding value estimated by the i-th particle. $P(E^l)$ is the discrete probability distribution of the estimation error of the vehicle accumulation on road segment l.

5. Simulation study

5.1. Experimental setup

To test the framework we consider the area shown in Fig. 5, which consists of two signalized intersections and six sensors (represented by green bars) that can detect individual vehicle passages. Each traffic light (*TLB* and *TLD*) has a fixed cycle time of 60 s, and the splits for red and green are 30 s each (as shown in the right part of Fig. 5). The vehicle arrivals obey a Poisson process, and the mean of inter-arrival time is 6 s (undersaturated condition). When generating a vehicle, the probabilities that it goes to Sink 1, Sink 2 and Sink 3 are 60%,20% and 20%, respectively.

In this paper, we use the Intelligent Driver Model (IDM) (Treiber et al., 2000) as our car-following model. The IDM defines a vehicle's acceleration as

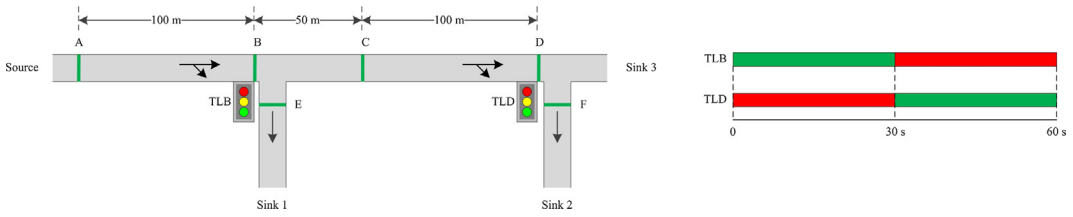


Fig. 5. A road stretch with two signalized intersections.

$$\dot{v}_{IDM}(s,v,\Delta v) = a\left\{1-\left(\frac{v}{v_0}\right)^4-\left(\frac{s^*(v,\Delta v)}{s}\right)^2\right\}$$
$$s^*(v,\Delta v) = s_0 + vT + \frac{v\Delta v}{2\sqrt{ab}}$$

where the parameters are assigned with typical values for city traffic (Treiber and Kesting, 2013), which are listed in Table 1.

The objective is to apply the proposed data assimilation framework to reconstruct the trajectories of vehicles entering from A and leaving from D. To this end, we run the simulation for 1800 s (30 cycles), and record (ground truth) vehicle trajectories at 1 Hz along with (ground truth) passage times across each sensor. We allow for a warm up time of 240 s, and use 14 cycles (from $t = 240$ s to $t = 1080$ s) for study, in which the passages from each sensor were processed into different datasets with different detection accuracy (p) and occurrence rate of over-count (λ), in order to imitate different sensor qualities. Travel time is measured every 3 min. We set the measurement interval to $\Delta T = 60$ seconds, i.e. passage times become available every 60 s.

5.2. Evaluation criteria

We assess the data assimilation framework in line with the two aspects shown in Fig. 1: correctness of the reconstructed vehicle passages at each sensor; and agreement of the reconstructed vehicle trajectories with the ground truth.

5.2.1. Vehicle passages and departure times per cycle and over multiple cycles

Let $E_i^j = \{t_1, t_2, \ldots, t_{n_i}\}$ and $\widehat{E}_i^j = \{\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_{\hat{n}_i}\}$ depict the ground truth and estimated passing times at sensor j during the i-th cycle respectively. We define the estimation error of the number of vehicle passages as

$$E^j_{number,i} = 100\% \times (n_i - \hat{n}_i)/n_i,$$

and the departure time error as

$$E^j_{match,i} = 100\% \times (n_i - \hat{n}_i^m)/n_i,$$

in which $\hat{n}_i^m$ is the number of time instants in E_i^j that are *matched* with certain time instants in $\widehat{E}_i^j$ using the match procedure introduced in Section 4.4. In this procedure we use a threshold value for the minimum time headway h_{min} to determine which ground truth and estimated departure times most likely coincide. $E^j_{match,i}$ essentially defines the percentage of passages in E_i^j that are *not accurately* reconstructed.

Over multiple cycles we take the mean absolute percent error, that is,

$$\overline{E}^j_{number} = \frac{1}{n_{cycles}}\sum_{i=1}^{n_{cycles}} |E^j_{number,i}|, \overline{E}^j_{match} = \frac{1}{n_{cycles}}\sum_{i=1}^{n_{cycles}} |E^j_{match,i}| \tag{15}$$

5.2.2. Generalized flow and density

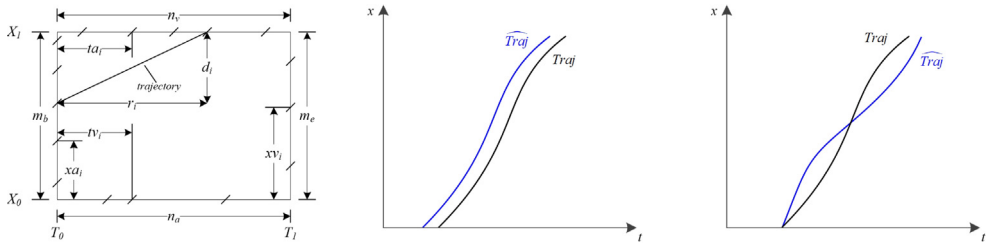
To assess the quality of vehicle trajectory reconstruction at the macroscopic level, we compare the generalized flow q and density k for a time–space region using Edie's well known definitions (Edie, 1963):

$$q = \frac{\sum_i d_i}{XT}, k = \frac{\sum_i r_i}{XT},$$

which, using the variables in Fig. 6a, can be computed as

Table 1
The parameters for the IDM.

Name	Maximum acceleration a (m/s^2)	Comfortable deceleration b (m/s^2)	Minimum distance s_0 (m)	Safe time headway T (s)	Desired speed v_0 (m/s)
Value	1.0	1.5	2.0	1.0	15.0



(a) Macroscopic agreement (b) Translation error only (c) Translation and distortion error

Fig. 6. Evaluation of vehicle trajectory reconstruction.

$$q = \frac{n_v X - \sum_{i=1}^{m_b} xa_i + \sum_{i=1}^{m_e} xv_i}{XT}$$
$$k = \frac{m_e T - \sum_{i=1}^{n_a} ta_i + \sum_{i=1}^{n_v} tv_i}{XT}, \tag{16}$$

where n_a is the number of arrivals at cross-section X_0,n_v is the number of departures at cross-section X_1,m_b is the vehicle accumulation at T_0, and m_e is the vehicle accumulation at T_1. The corresponding estimation errors for flow and density in the i-th cycle are defined as percent errors:

$$E_{flow,i} = 100\% \times (q_i - \hat{q}_i)/q_i$$
$$E_{density,i} = 100\% \times (k_i - \hat{k}_i)/k_i$$

where q_i,k_i are the ground truth values of flow and density in the i-th cycle, respectively, and $\hat{q}_i$,$\hat{k}_i$ are their corresponding estimated values. Finally, again over multiple cycles we take the mean absolute percent error, that is,

$$\overline{E}_{flow} = \frac{1}{n_{cycles}} \sum_{i=1}^{n_{cycles}} |E_{flow,i}|, \overline{E}_{density} = \frac{1}{n_{cycles}} \sum_{i=1}^{n_{cycles}} |E_{density,i}| \tag{17}$$

5.2.3. Agreement between estimated and ground-truth trajectories

To assess how well *a single* reconstructed vehicle trajectory $\widehat{Traj}$ fits to the ground truth *Traj*, we consider two error measures that capture the error in location and the error in (local) speed, that we term translation and distortion error respectively. The so called *translation error* quantifies how much effort is needed to shift *Traj* along the time axis to obtain $\widehat{Traj}$. The less effort, the better the estimation. We define it as

$$E_{tr}^{t}(Traj, \widehat{Traj}) = \frac{1}{N_{\delta x}} \sum_{i=1}^{N_{\delta x}} \frac{|t(x_i) - \hat{t}(x_i)|}{TT(Traj)} \times 100\% \tag{18}$$

where $t(x)$,$\hat{t}(x)$ are the time instants when trajectory *Traj* and its estimation $\widehat{Traj}$ cross location x, respectively; $N_{\delta x}$ is the number of discretized locations that *Traj* and $\widehat{Traj}$ crosses; $TT(Traj)$ is the total travel time from the upstream to the downstream, measured from the ground truth trajectory *Traj*.

The so-called *distortion error* quantifies how much effort is needed to distort *Traj* to obtain $\widehat{Traj}$. The less effort, the better the estimation. This distortion is the result of errors in speed along the trajectory. We define it as

$$E_{tr}^{v}(Traj, \widehat{Traj}) = \frac{1}{N_{\delta x}} \sum_{i=1}^{N_{\delta x}} \frac{|v(x_i) - \hat{v}(x_i)|}{\bar{v}(Traj)} \times 100\% \tag{19}$$

where $v(x)$,$\hat{v}(x)$ are the speeds when trajectory *Traj* and its estimation $\widehat{Traj}$ cross location x, respectively; $\bar{v}(Traj)$ is the mean speed measured from the ground truth trajectory *Traj* (i.e., total traveled distance divided by total travel time within the time–space region).

Fig. 6b illustrates a case where a trajectory is perfectly estimated safe for a translation error. Fig. 6c shows a case in which there is a distortion error (and by implication also a translation error). We finally also consider an overall error which combines the two effects (we arbitrarily give both error components equal weight):

$$E_{tr}^{t,v}(Traj, \widehat{Traj}) = \frac{E_{tr}^{t}(Traj, \widehat{Traj}) + E_{tr}^{v}(Traj, \widehat{Traj})}{2} \tag{20}$$

Finally, we define error measures for two *sets of trajectories*. Prior to definition, we need to figure out which trajectory in the set of ground truth trajectories $TS = \{Traj_i\}$ is estimated by which trajectory in the set of reconstructed trajectories $\widehat{TS} = \{\widehat{Traj}_j\}$. To this end, we denote the start point of trajectory $Traj_i \in TS$ as $(t_{0,i}, x_{0,i})$, and for a trajectory $\widehat{Traj}_j \in \widehat{TS}$, as $(\hat{t}_{0,j}, \hat{x}_{0,j})$. If the two start points are *close enough*, $\widehat{Traj}_j$ is regarded as the estimation of $Traj_i$. By 'close enough', we mean that one of the following conditions is met:

- $x_{0,i} = \hat{x}_{0,j} = X_0 \cap |t_{0,i} - \hat{t}_{0,j}| < h_{min}$ or
- $t_{0,i} = \hat{t}_{0,j} \cap |x_{0,i} - \hat{x}_{0,j}| < s_{min}$,

where h_{min} and s_{min} are the minimum time headway and the minimum space headway, respectively. As a result we now define the following three error measures for set $TS = \{Traj_i\}$ being estimated by set $\widehat{TS} = \{\widehat{Traj_j}\}$:

$$\begin{aligned}
\overline{E}^{t}_{tr}(TS,\widehat{TS}) &= \frac{1}{n}\sum E^{t}_{tr}(Traj_i,\widehat{Traj}_{j_i}) \\
\overline{E}^{v}_{tr}(TS,\widehat{TS}) &= \frac{1}{n}\sum E^{v}_{tr}(Traj_i,\widehat{Traj}_{j_i}) \\
\overline{E}^{t,v}_{tr}(TS,\widehat{TS}) &= \frac{1}{n}\sum E^{t,v}_{tr}(Traj_i,\widehat{Traj}_{j_i})
\end{aligned} \tag{21}$$

where $\widehat{Traj}_{j_i} \in \widehat{TS}$ is the estimation of $Traj_i \in TS$, and n is the number of vehicle trajectories in TS which have corresponding estimations in $\widehat{TS}$.

Finally note that since $\bar{v}(Traj)$ appears in the denominator of Eq. (19), both the distortion error (Eq. (19)) and the overall error (Eq. (20)) are sensitive to very low speeds. Therefore, the distortion error is not informative when average speeds are small, e.g. in oversaturated traffic conditions. We will return to this point further below (Section 5.4).

5.3. Results

In this section, we use a fixed detection accuracy of $p = 0.9$; an occurrence rate of over-counts $\lambda = 1/300\ \text{s}^{-1}$; and we assume that travel time observations are available every 3 min. Due to space limitations, we restrict the presentation to road stretch AB in Fig. 5.

5.3.1. Reconstructed vehicle passages per cycle

The vehicle accumulation estimation results for the roadstretch between sensors A and B are shown in Fig. 7, where the left plot shows the estimated vehicle accumulation, while the right plot depicts the associated histogram of—indeed approximately zero-mean!—estimation errors. Fig. 8 shows the ground-truth, observed, estimated and matched numbers of vehicle passages at upstream sensor (A) and downstream sensor (B) for a sequence of 14 cycles, in blue, red, green, and yellow, respectively.

Table 2 lists the average errors (Eq. (15)) over 14 cycles, which are 7.43% and 5.60% for sensor A; and 2.70% and 5.63% for sensor B, respectively. This implies that our method is able to reconstruct ca. 95% of the vehicle passages, which is a promising result, considering our dataset contains around 10% miss-counts and over-counts ($p = 0.9, \lambda = 1/300\ \text{s}^{-1}$).

5.3.2. Generalized flow and density

The estimation results for flow q and density k on roadstretch AB are shown in Fig. 9a and b, respectively. The average estimation error over 14 cycles (computed with Eq. (17)) for flow q and density k are 5.26% and 5.23%, respectively.

It is insightful to analyze a few results in detail. For example, flow q and density k are overestimated in the 6-th cycle (300–360 s) while underestimated in the 13-th cycle (720–780 s). The (large) error in the 6-th cycle is mainly a result of the errors in the vehicle accumulation. As shown in the left plot in Fig. 7, vehicle accumulation is overestimated in the end of the cycle ($t = 360$ s). Based on the definitions of q and k in Eq. (16), these errors will lead to overestimations of q and k. In the 13-th cycle, the (large) error is probably due to the miss-counts of sensor A, as shown in Fig. 10. The vehicles missed at sensor A are also not fully observed at downstream sensors. In this case, the chance to successfully reconstruct their trajectories is very low. Consequently, the flow q and the density k will be underestimated because the missed vehicles do not show up at sensor B (n_v thus has a smaller value in the computation of q) and their travel times are not included in the computation of k as shown in Eq. (16).

In the 7-th cycle (360–420 s), the flow q is largely underestimated, which is mainly due to the overestimation of vehicle accumulation at $t = 360$ s. This overestimation in vehicle accumulation leads to a smaller number of vehicle arrivals upstream (sensor A)

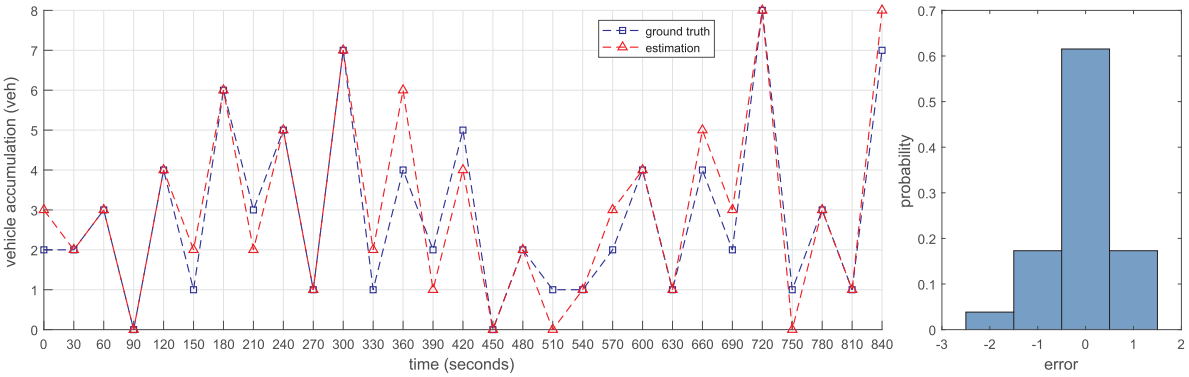
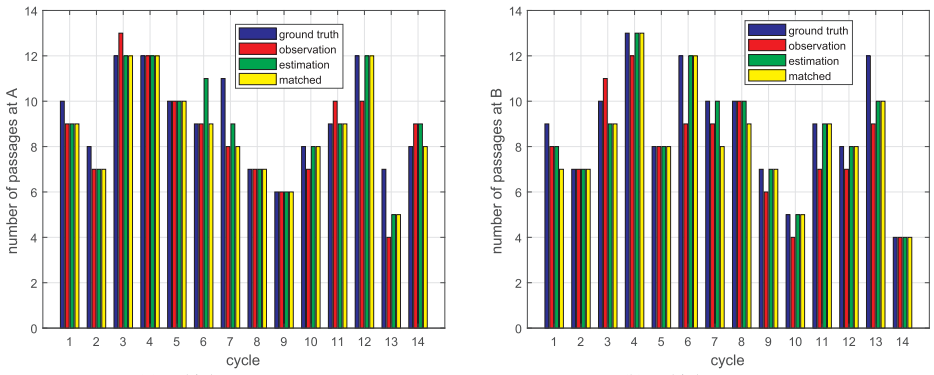


Fig. 7. The vehicle accumulation estimation results for the roadstretch between sensors A and B using the correction method (the left plot shows the estimated vehicle accumulation, while the right plot depicts the histogram of the estimation errors).

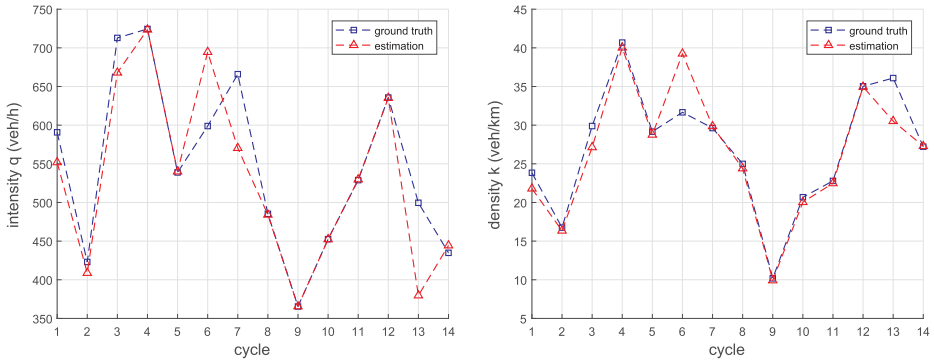


(a) Vehicle passages at A (b) Vehicle passages at B

Fig. 8. Number of vehicle passages per cycle.

Table 2
The estimation error of the number of vehicle passages and the percentage of passages that are not accurately reconstructed as defined in Eq. (15) computed over 14 cycles at sensors A and B respectively.

	$\overline{E}^{A}_{number}$	$\overline{E}^{A}_{match}$	$\overline{E}^{B}_{number}$	$\overline{E}^{B}_{match}$
Performance	7.43%	5.60%	2.70%	5.63%



(a) The estimated flow q (b) The estimated density k

Fig. 9. The estimated flow q and density k on roadstretch AB

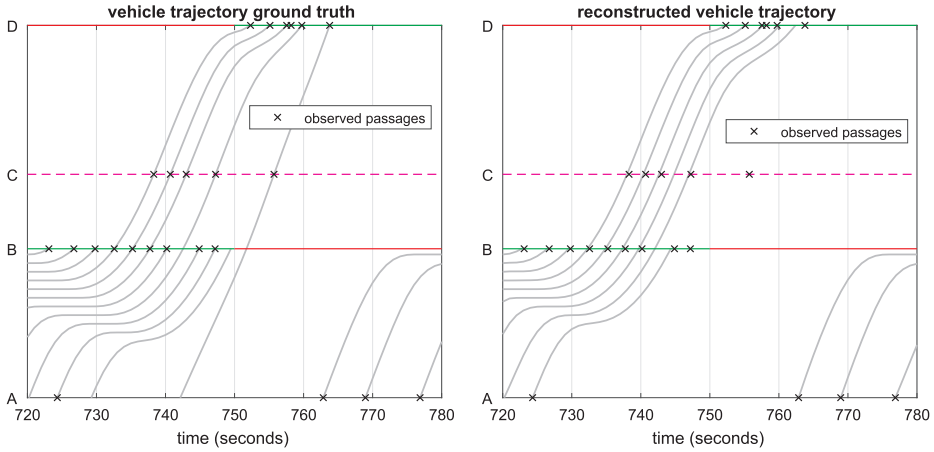


Fig. 10. Vehicle trajectories in the 13-th cycle.

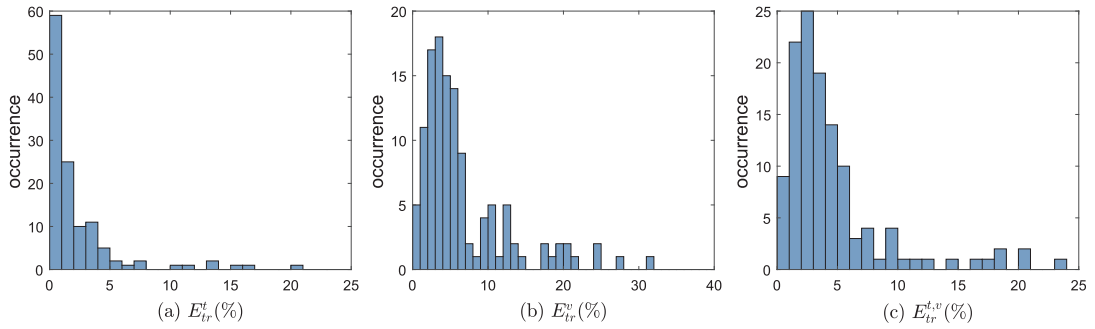


Fig. 11. The histogram of translation error E^t_{tr}, distortion error E^v_{tr}, and overall error $E^{t,v}_{tr} = (E^t_{tr} + E^v_{tr})/2$ (123 pairs of trajectories in total; the width of each bin is 1.0%).

in order for the simulated passages to match the observed passages downstream (sensor *B*). As a consequence, q is assigned a smaller value (see the definition in Eq. (16)).

5.3.3. Translation error and distortion error

In the time–space region on roadstretch *AB*, i.e., [0 s,840 s] × [0 m,100 m], there are 131 vehicle trajectories in the ground truth dataset; our method reconstructs 128 trajectories in the same region, and 123 of them can be matched with trajectories in the ground truth dataset. The translation error E^t_{tr} (Eq. (18)), distortion error E^v_{tr} (Eq. (19)), and overall error $E^{t,v}_{tr}$ (Eq. (20)) for each pair of trajectories (123 pairs) are computed, and the histograms of these errors are depicted in Fig. 11(a), (b), and (c), respectively. The average values for the translation error, the distortion error, and the overall error are $\overline{E}^t_{tr} = 2.56\%$,$\overline{E}^v_{tr} = 6.98\%$, and $\overline{E}^{t,v}_{tr} = 4.77\%$, respectively. Again these averages of about 5% are promising given the degree of sensor errors we imposed. Since the overall error $E^{t,v}_{tr}$ is a linear combination of E^t_{tr} and E^v_{tr}, its histogram follows the shape of the histogram of E^v_{tr}.

We make two general observations based on the results. First, the histograms in Fig. 11 do illustrate that some pairs of trajectories have large errors (15% and more). These pairs of trajectories mainly lie in the 6-th cycle (300–360 s) and the 7-th cycle (360–420 s), which are depicted in Fig. 12. As shown in the left plot in Fig. 7, the corresponding vehicle accumulation is overestimated in the end of the 6-th cycle ($t = 360$ s). This results in the (false) reconstruction of two trajectories (originating from $t \approx 330$ s and $t \approx 340$ s) in order to match the vehicle accumulation at $t = 360$ s. Essentially the method falsely reconstructs a queue, yielding strongly sloped trajectories after $t \approx 350$ s until that queue is resolved. This discrepancy in trajectory shape leads to a large translation and distortion error.

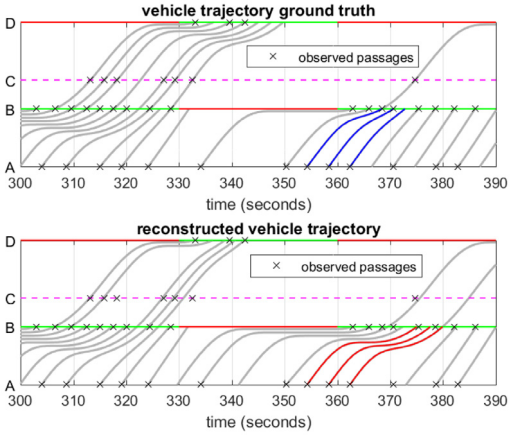
Second, and related to this point, the results indicate that the distortion errors are in general larger than the translation errors. This makes intuitively sense: the data (passages and accumulation) give direct information on how much vehicles we expect where and when. This information, however, can be reconstructed with an—in principle—infinite number of individual speed profiles. The simulation models applied constrain this large solution space to those trajectories that are plausible. We take the last pair of trajectories (originating from $t \approx 405$ s) in Fig. 12d as an example. We redraw this pair of trajectories in the left plot of Fig. 13, and we show their time differences (see Eq. (18)) in the middle of Fig. 13, and we depict their speed differences (see Eq. (19)) in the right plot of Fig. 13. The translation error and distortion error for this pair are $E^t_{tr} = 5.97\%$ and $E^v_{tr} = 31.16\%$, respectively. From Fig. 13, we can see that $|v(x) - \hat{v}(x)|/\overline{v}$ is less robust to a small trajectory discrepancy than $|t(x) - \hat{t}(x)|/TT$. For example, at $x \in [0,50]$ where the estimated trajectory is very close to the ground truth trajectory, the speed difference $|v(x) - \hat{v}(x)|/\overline{v}$ can reach above 20%, while time difference $|t(x) - \hat{t}(x)|/TT$ is only around 1%. Therefore, the distortion errors are in general larger than the translation errors. This also explains why the translation errors are not as large as the distortion errors for the last three pairs of trajectories in Fig. 12d.

5.4. Oversaturated traffic conditions

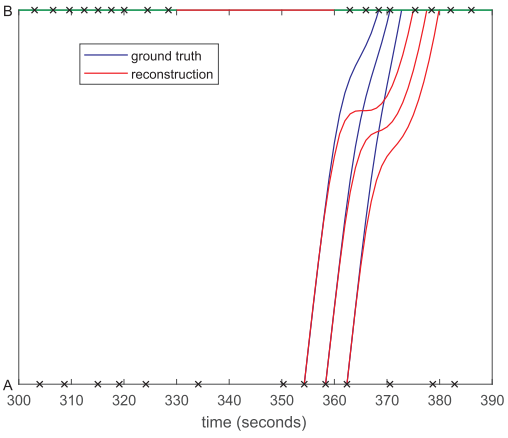
We also tested the data assimilation framework in oversaturated traffic conditions, in which the mean of inter-arrival time is set to 4 s. The results (obtained from 10 independent runs) are shown in Table 3. As expected, we see very high distortion errors (and thus large overall errors), since the average speed is very small in oversaturated conditions. However, since the traffic shows less variability (thus less uncertainty) in oversaturated conditions, on each of the other error measures we find similar or even better results when comparing with those (obtained from a single run in Section 5.3) in undersaturated conditions, which are 7.43% ($\overline{E}^A_{number}$), 5.60% ($\overline{E}^A_{match}$), 2.70% ($\overline{E}^B_{number}$), 5.63% ($\overline{E}^B_{match}$), 5.26% ($\overline{E}_{flow}$), 5.23% ($\overline{E}_{density}$), 2.56% ($\overline{E}^t_{tr}$), 6.98% ($\overline{E}^v_{tr}$), and 4.77% ($\overline{E}^{t,v}_{tr}$), respectively.

6. Sensitivity analysis and extensions

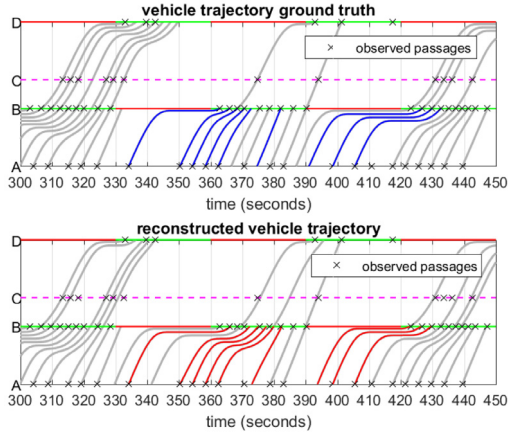
In this section, we test the framework in undersaturated traffic conditions (mean of inter-arrival time is 6 s) by varying with several factors which will likely influence the data assimilation results. These factors include sensor data quality, modeling errors, the number of particles employed. We also discuss a few possible extensions to the framework. For each set of parameters, we run the experiment 10 times with different random seeds.



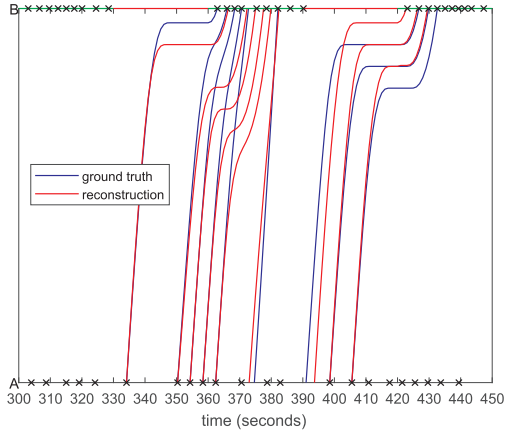
(a) General view of trajectories whose $E_{tr}^{t} > 15\%$



(b) Enlarged view of trajectories whose $E_{tr}^{t} > 15\%$



(c) General view of trajectories whose $E_{tr}^{v} > 15\%$



(d) Enlarged view of trajectories whose $E_{tr}^{v} > 15\%$

Fig. 12. The trajectories whose translation/distortion error is larger than 15%.

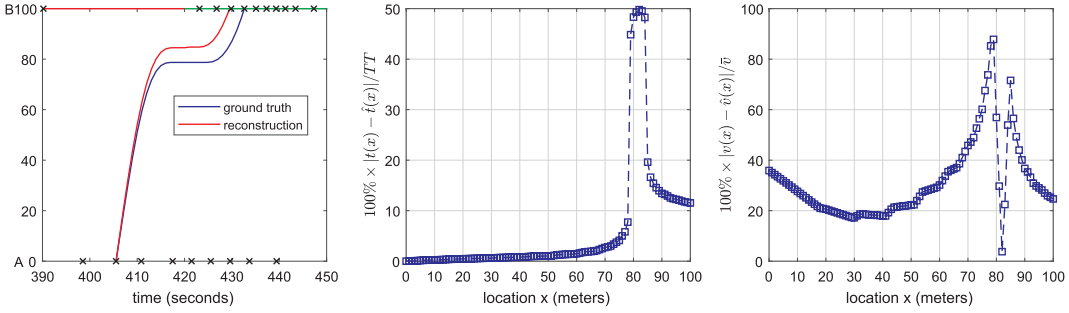


Fig. 13. Time differences and speed differences of a pair of trajectories (the plot in the left shows the pair of trajectories, the plot in the middle depicts the time series of $100\% \times |t(x) - \hat{t}(x)|/TT$, and the plot in the right depicts the time series of $100\% \times |v(x) - \hat{v}(x)|/\bar{v}$; for this pair of trajectories, $E_{tr}^{t} = 5.97\%$,$E_{tr}^{v} = 31.16\%$).

6.1. Sensitivity analysis

6.1.1. The difference between estimated and error-free vehicle accumulations

Vehicle accumulations are key in the trajectory reconstruction method, but difficult to come by. In this paper, we used a correction method to estimate vehicle accumulations through cumulative counts and coarsely available travel time observations. Although this method yields good results, the estimates are certainly not error free. Table 4 summarizes the results when error-free vehicle accumulations are assimilated. Although the original results (top row in Table 4) are acceptable and around 10% or less, these

Table 3
The data assimilation results in oversaturated traffic conditions (mean of inter-arrival time is 4 s). For all simulations we have $p = 0.9, \lambda = 1/300\ s^{-1}, N_p = 1000$ (i.e. 1000 particles). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

Traffic conditions	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}^A_{number}$ (%)	$\overline{E}^A_{match}$ (%)	$\overline{E}^B_{number}$ (%)	$\overline{E}^B_{match}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}^t_{tr}$ (%)	$\overline{E}^v_{tr}$ (%)	$\overline{E}^{t,v}_{tr}$ (%)
Oversaturation	8.33 (8.33, 9.62)	9.62 (8.97, 9.62)	0.00 (0.00, 0.00)	8.97 (8.97, 9.62)	2.30 (1.60, 2.78)	5.99 (4.82, 7.05)	4.92 (3.76, 5.24)	53.41 (45.38, 59.63)	29.17 (24.57, 32.43)

Table 4
Data assimilation results when accurate vehicle accumulations are available. For all simulations we have $p = 0.9, \lambda = 1/300\ s^{-1}, N_p = 1000$ (i.e. 1000 particles). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

Estimated accumulations	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}^A_{number}$ (%)	$\overline{E}^A_{match}$ (%)	$\overline{E}^B_{number}$ (%)	$\overline{E}^B_{match}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}^t_{tr}$ (%)	$\overline{E}^v_{tr}$ (%)	$\overline{E}^{t,v}_{tr}$ (%)
Inaccurate accumulations	9.37 (7.70, 11.18)	10.47 (7.81, 11.02)	5.90 (5.09, 6.64)	9.97 (7.15, 10.18)	8.37 (7.17, 9.65)	9.48 (7.64, 11.84)	3.57 (3.14, 3.73)	10.49 (10.16, 11.05)	7.04 (6.77, 7.37)
Error-free accumulations	5.84 (5.20, 6.86)	7.71 (6.56, 8.45)	5.09 (4.43, 6.41)	6.62 (5.74, 7.14)	6.65 (6.22, 7.23)	7.36 (6.95, 8.10)	2.19 (2.01, 2.27)	9.39 (8.75, 9.84)	5.76 (5.48, 6.20)

results confirm that indeed all error measures (number of passages, departure times, flow, density and the three trajectory errors, respectively) decrease significantly, also in the statistical sense. By investing in vehicle counting (better loops, or via radar or other sensing devices), around 40% more accurate cumulative counts ($\overline{E}^j_{number}, \overline{E}^j_{match} j \in \{A,B\}$); around 25–30% more accurate macroscopic estimates ($\overline{E}_{flow}, \overline{E}_{density}$); and also more accurate microscopic vehicle dynamics (trajectories) can be estimated, particularly when it comes to the correct location of the trajectories (38% improvement in $\overline{E}^t_{tr}$ versus 10% improvement in $\overline{E}^v_{tr}$ respectively).

6.1.2. Effect of sampling rate of travel time observations

The experiment in Section 5 assumes that the travel time observations are available every 3 min. In this section, we explore how the sampling rate of the travel time observations affect the data assimilation results. This effect is essentially predictable. On the one hand, van Lint and Hoogendoorn (2015) shows that more frequent travel time observations leads to more accurate vehicle accumulation estimations. On the other hand, the results in Table 4 reveal that with more accurate vehicle accumulations, we can reconstruct more accurate vehicle trajectories. Therefore, we can predict that with more frequent travel time observations, the data assimilation results will be improved. This prediction is confirmed by the experiment results shown in Table 5, which tell that the trajectory reconstruction results become more accurate as the travel time observations are fed more frequently, but this trend is below proportional.

Table 5
The influence of the sampling rate of the travel time observations. For all simulations we have $p = 0.9, \lambda = 1/300\ s^{-1}, N_p = 1000$ (i.e. 1000 particles). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

Sampling rate (minutes)	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}^A_{number}$ (%)	$\overline{E}^A_{match}$ (%)	$\overline{E}^B_{number}$ (%)	$\overline{E}^B_{match}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}^t_{tr}$ (%)	$\overline{E}^v_{tr}$ (%)	$\overline{E}^{t,v}_{tr}$ (%)
9	11.21 (10.35, 11.94)	11.48 (10.31, 13.98)	7.43 (6.16, 8.13)	11.06 (9.87, 11.86)	10.40 (9.65, 10.88)	14.11 (11.21, 17.03)	4.49 (3.81, 4.82)	14.37 (12.25, 15.50)	9.52 (8.00, 10.16)
6	10.00 (9.27, 11.62)	11.50 (10.75, 11.92)	7.91 (6.14, 9.20)	10.51 (9.81, 12.58)	9.50 (8.22, 10.33)	15.19 (10.85, 18.44)	3.92 (3.56, 4.20)	12.45 (11.90, 12.91)	8.22 (7.60, 8.56)
3	9.37 (7.70, 11.18)	10.47 (7.81, 11.02)	5.90 (5.09, 6.64)	9.97 (7.15, 10.18)	8.37 (7.17, 9.65)	9.48 (7.64, 11.84)	3.57 (3.14, 3.73)	10.49 (10.16, 11.05)	7.04 (6.77, 7.37)

Table 6
The influence of p on the data assimilation results ($\lambda = 1/\infty$ s^{-1},$N_p = 1000$). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

p	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}^{A}_{number}$ (%)	$\overline{E}^{A}_{match}$ (%)	$\overline{E}^{B}_{number}$ (%)	$\overline{E}^{B}_{match}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}^{t}_{tr}$ (%)	$\overline{E}^{v}_{tr}$ (%)	$\overline{E}^{t,v}_{tr}$ (%)
0.6	14.15 (12.85, 15.01)	17.67 (16.89, 18.49)	6.65 (6.51, 7.69)	18.32 (16.34, 19.39)	11.96 (10.49, 13.31)	19.77 (17.88, 21.43)	7.49 (6.79, 8.75)	16.05 (15.52, 17.64)	11.94 (11.33, 12.97)
0.7	10.64 (8.94, 12.11)	12.12 (10.48, 15.49)	7.93 (5.25, 10.86)	13.21 (10.60, 14.54)	8.00 (7.39, 8.70)	12.22 (10.96, 16.42)	5.80 (5.37, 6.91)	16.10 (14.01, 17.64)	11.15 (9.59, 11.89)
0.8	7.99 (6.45, 9.45)	7.99 (7.71, 9.09)	5.55 (4.19, 7.36)	7.85 (7.37, 9.37)	7.10 (5.60, 7.53)	12.54 (10.00, 15.30)	4.72 (4.27, 5.06)	13.19 (11.97, 14.49)	8.72 (8.36, 9.78)
0.9	5.71 (4.42, 7.64)	2.96 (2.10, 3.35)	6.87 (6.44, 7.20)	3.73 (2.28, 4.12)	6.29 (4.88, 7.09)	12.11 (11.09, 14.32)	3.20 (2.37, 3.97)	12.18 (10.99, 12.86)	7.67 (6.68, 8.68)
1.0	0.00 (0.00, 0.00)	0.00 (0.00, 0.00)	0.00 (0.00, 0.00)	0.79 (0.00, 0.79)	0.30 (0.13, 0.68)	1.38 (1.31, 1.43)	1.24 (1.21, 1.27)	4.95 (4.84, 5.28)	3.09 (3.02, 3.28)

6.1.3. Effect of sensor quality

In this paper, the sensor quality is characterized by two parameters: detection accuracy p and occurrence rate of over-count λ. When varying one of the two, the other is assigned a perfect value, i.e. when varying p, we set λ to $1/\infty$ s^{-1} (no over-counts); when varying λ, we set $p = 1$ (no miss-counts). For all experiments we set $N_p = 1000$. The results are shown in Tables 6 and 7, respectively. The results are in line with our expectations that the performance improves as the sensors become more accurate. We can conclude that the proposed method is quite robust to data errors. Even with 40% miss-counts or one over-count every two minutes, all estimation errors are (well) within a 20% range.

6.1.4. Effect of model errors

In previous sections, the model we used to carry out data assimilation is the same with that we used to generate the ground truth data, which implies that we have a perfect model of the reality. This is a very strong assumption. In this section, we investigate the data assimilation results in case the model has errors. Specifically, we test two cases: (1) assimilating data generated by the IDM using the IDM with 5% calibration errors on the minimum spacing s_0 and the desired time headway T; (2) assimilating data generated by the IDM using a different car-following model, the Improved Full Velocity Difference Model (IFVDM) (Treiber and Kesting, 2013), which defines a vehicle's acceleration as

$$v_{opt}(s) = max\left\{0, v_0 \frac{\tanh(\frac{s}{\Delta s} - \beta) + \tanh\beta}{1 + \tanh\beta}\right\}$$

$$\dot{v}_{IFVDM} = \frac{v_{opt}(s) - v}{\tau} - \frac{\gamma \Delta v}{max\{1, s/(v_0 T)\}}$$

Table 7
The influence of λ on the data assimilation results ($p = 1.0$,$N_p = 1000$). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

$\lambda(s^{-1})$	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}^{A}_{number}$ (%)	$\overline{E}^{A}_{match}$ (%)	$\overline{E}^{B}_{number}$ (%)	$\overline{E}^{B}_{match}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}^{t}_{tr}$ (%)	$\overline{E}^{v}_{tr}$ (%)	$\overline{E}^{t,v}_{tr}$ (%)
1/120	10.33 (9.21, 13.14)	12.17 (11.05, 14.34)	10.08 (8.93, 12.61)	12.60 (10.76, 14.26)	10.64 (8.89, 12.44)	17.83 (16.59, 19.78)	3.78 (3.52, 4.00)	13.25 (12.73, 15.28)	8.42 (8.25, 9.79)
1/180	7.94 (6.39, 9.33)	8.57 (7.17, 11.14)	6.81 (5.58, 8.66)	8.24 (6.90, 9.99)	7.49 (5.85, 9.09)	13.25 (11.18, 17.10)	3.45 (2.97, 3.80)	12.08 (9.67, 13.37)	7.81 (6.32, 8.51)
1/240	6.79 (5.88, 7.31)	6.79 (5.88, 7.31)	6.19 (5.73, 6.92)	6.83 (6.23, 7.44)	6.91 (6.36, 7.78)	14.92 (14.38, 18.16)	3.11 (2.94, 3.24)	11.99 (10.85, 12.76)	7.56 (6.89, 8.15)
1/300	4.59 (4.37, 6.58)	4.19 (3.67, 6.00)	5.05 (4.52, 5.91)	6.01 (4.79, 6.90)	5.03 (4.56, 6.09)	8.11 (6.29, 10.21)	2.35 (2.05, 2.52)	9.21 (7.51, 9.98)	5.83 (4.82, 6.22)
1/∞	0.00 (0.00, 0.00)	0.00 (0.00, 0.00)	0.00 (0.00, 0.00)	0.79 (0.00, 0.79)	0.30 (0.13, 0.68)	1.38 (1.31, 1.43)	1.24 (1.21, 1.27)	4.95 (4.84, 5.28)	3.09 (3.02, 3.28)

Table 8
The models used to assimilate the data generated by the IDM (a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.0 m, T = 1.0 s, v_0 = 15.0 m/s).

Model	Parameters
IDM (benchmark)	a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.0 m, T = 1.0 s, v_0 = 15.0 m/s
IDM (s_0−5%)	a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 1.9 m, T = 1.0 s, v_0 = 15.0 m/s
IDM (s_0 + 5%)	a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.1 m, T = 1.0 s, v_0 = 15.0 m/s
IDM (T−5%)	a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.0 m, T = 0.95 s, v_0 = 15.0 m/s
IDM (T + 5%)	a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.0 m, T = 1.05 s, v_0 = 15.0 m/s
IFVDM	Δs = 8.0 m, β = 1.5, v_0 = 15.0 m/s, τ = 5.0 s, T = 1.2 s, γ = 0.6 s^{-1}

where Δs is the transition width, β is the form factor, v_0 is the desired speed, τ is the adaption time, T is the time gap, and γ is the speed difference sensitivity. These parameters are assigned with the typical values of city traffic (Treiber and Kesting, 2013) as shown in the last row in Table 8. The parameters of the IDM which generates the ground truth data are a = 1.0 m/s^2, b = 1.5 m/s^2, s_0 = 2.0 m, T = 1.0 s, v_0 = 15.0 m/s. The results are summarized in Table 9.

The results in Table 9 reveal that the proposed method is robust with respect to model errors, although with the two cases involved, we cannot claim to have tested this exhaustively. In the case that the model has 5% calibration errors, the overall performance is not significantly different with a perfect model; in the case that a different model is applied, the performance does not degenerate except for the distortion error $\overline{E}_{tr}^{v}$, which is likely a result of different acceleration processes between the two models. Clearly, this will result in larger differences in individual speed dynamics and thus larger distortion errors. Since the overall error $E_{tr}^{t,v}$ is a linear combination of E_{tr}^{t} and E_{tr}^{v}, we also observe large error in the last column ($\overline{E}_{tr}^{t,v}$) of the last row in Table 9. This result does emphasize an important underlying point. Clearly, unless we have actual evidence (data), either macroscopically in the form of queue dynamics, or (even better) microscopically in the form of sample trajectories, the quality of the reconstruction completely depends on the validity of the microscopic models used in the framework for the specific case. Further below we show that such evidence can be naturally incorporated in the framework as well, and that it indeed improves the results.

6.1.5. *Effect of number of particles*

The influence of the number of particles (N_p) employed on the data assimilation results is summarized in Table 10. As expected, the overall performance has an upward tendency as the number of particles increases. With more particles, the proposed method can explore more possibilities on the entering times & speeds of individual vehicles and vehicle dispersion in the network, which leads to a better coverage of the system state space.

The good news, however, is that the trend is far below proportional, which becomes apparent when looking at results in a graph,

Table 9
The influence of model errors on the data assimilation results (p = 0.9,λ = 1/300 s^{-1},N_p = 1000). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25th and 75th percentiles.

Model type	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\overline{E}_{number}^{A}$ (%)	$\overline{E}_{match}^{A}$ (%)	$\overline{E}_{number}^{B}$ (%)	$\overline{E}_{match}^{B}$ (%)	$\overline{E}_{flow}$ (%)	$\overline{E}_{density}$ (%)	$\overline{E}_{tr}^{t}$ (%)	$\overline{E}_{tr}^{v}$ (%)	$\overline{E}_{tr}^{t,v}$ (%)
IDM (benchmark)	9.37 (7.70, 11.18)	10.47 (7.81, 11.02)	5.90 (5.09, 6.64)	9.97 (7.15, 10.18)	8.37 (7.17, 9.65)	9.48 (7.64, 11.84)	3.57 (3.14, 3.73)	10.49 (10.16, 11.05)	7.04 (6.77, 7.37)
IDM (s_0−5%)	8.86 (8.57, 10.24)	10.06 (9.50, 11.05)	6.20 (5.41, 6.41)	9.52 (8.02, 10.92)	7.65 (6.96, 8.86)	10.67 (9.22, 12.08)	3.69 (3.27, 3.84)	11.29 (10.53, 12.34)	7.51 (6.98, 7.81)
IDM (s_0 + 5%)	9.31 (8.74, 10.45)	9.49 (7.38, 10.41)	5.67 (4.94, 6.42)	8.94 (7.66, 9.63)	8.42 (7.14, 8.91)	8.67 (7.67, 9.85)	2.93 (2.51, 3.65)	10.40 (9.54, 11.45)	6.77 (6.09, 7.37)
IDM (T−5%)	9.45 (8.69, 10.05)	10.31 (8.35, 10.60)	5.69 (5.36, 7.41)	9.29 (8.10, 9.47)	8.01 (7.48, 9.29)	9.83 (8.23, 10.90)	3.41 (2.95, 3.63)	10.93 (9.34, 12.57)	7.27 (6.24, 8.10)
IDM (T + 5%)	9.64 (8.53, 11.38)	9.98 (9.56, 11.73)	6.82 (5.65, 7.61)	9.69 (8.88, 11.11)	8.32 (7.90, 9.48)	10.77 (10.20, 13.09)	3.60 (3.38, 4.09)	11.14 (10.70, 11.77)	7.46 (7.33, 7.74)
IFVDM	13.04 (10.53, 13.57)	11.79 (9.89, 13.24)	9.46 (8.99, 12.06)	12.49 (9.98, 13.65)	10.42 (9.00, 11.86)	14.06 (12.57, 15.62)	5.72 (5.59, 5.82)	30.66 (28.57, 31.20)	18.14 (17.12, 18.54)

Table 10
The influence of N_p on the data assimilation results ($p = 0.9, \lambda = 1/300\ \mathrm{s}^{-1}$). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

N_p	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\bar{E}^A_{number}$ (%)	$\bar{E}^A_{match}$ (%)	$\bar{E}^B_{number}$ (%)	$\bar{E}^B_{match}$ (%)	$\bar{E}_{flow}$ (%)	$\bar{E}_{density}$ (%)	$\bar{E}^t_{tr}$ (%)	$\bar{E}^v_{tr}$ (%)	$\bar{E}^{t,v}_{tr}$ (%)
100	12.03 (11.18, 12.64)	11.78 (10.58, 12.37)	7.83 (7.42, 9.28)	10.87 (10.51, 11.60)	10.60 (10.45, 10.71)	14.54 (12.79, 15.10)	4.17 (4.02, 4.37)	13.46 (12.83, 13.84)	8.78 (8.47, 9.12)
400	10.41 (10.13, 10.74)	10.35 (9.76, 11.21)	6.67 (6.32, 6.90)	9.61 (9.31, 10.12)	9.06 (8.79, 9.69)	11.73 (10.09, 13.49)	3.65 (3.44, 3.93)	11.99 (11.31, 12.19)	7.85 (7.26, 7.99)
700	9.52 (8.23, 10.42)	9.23 (8.47, 9.82)	5.79 (5.13, 6.41)	8.57 (7.47, 8.99)	7.81 (7.35, 8.67)	10.36 (9.37, 11.41)	3.38 (3.15, 3.51)	10.82 (10.34, 11.60)	6.94 (6.70, 7.44)
1000	9.37 (7.70, 11.18)	10.47 (7.81, 11.02)	5.90 (5.09, 6.64)	9.97 (7.15, 10.18)	8.37 (7.17, 9.65)	9.48 (7.64, 11.84)	3.57 (3.14, 3.73)	10.49 (10.16, 11.05)	7.04 (6.77, 7.37)
2000	7.87 (6.61, 9.10)	7.24 (6.52, 8.70)	4.84 (3.47, 5.88)	6.70 (6.21, 7.72)	7.09 (6.14, 7.55)	7.64 (6.25, 9.17)	3.03 (2.72, 3.24)	9.89 (9.52, 10.24)	6.57 (6.36, 6.73)

as presented in Fig. 14(a) by means of absolute numbers (similar to those in Table 10) and in Fig. 14(b) by dividing all the error measures by the corresponding performance at $N_p = 1000$ (the ensemble size chosen in this paper). What Fig. 14(b) shows, is that a reduction in ensemble size from $N_p = 1000$ to $N_p = 100$ (i.e. 1000%) leads to an increase in error metrics ranging from less than 10% (cumulative count at location *B*) to slightly over 50% (error in density); whereas doubling the ensemble size (to $N_p = 2000$) improves the performance no more than between 5 and 25%. Although relatively such gains may seem worthwhile, Fig. 14(a) shows the actual gains are small. In our view, the increased error measures in the case of $N_p = 100$ (Table 10 top row) are well within acceptable bounds; and it seems we could have safely decreased the number of particles in our experiment from $N_p = 1000$ to $N_p = 700$ without a significant loss of accuracy in terms of all error measures.

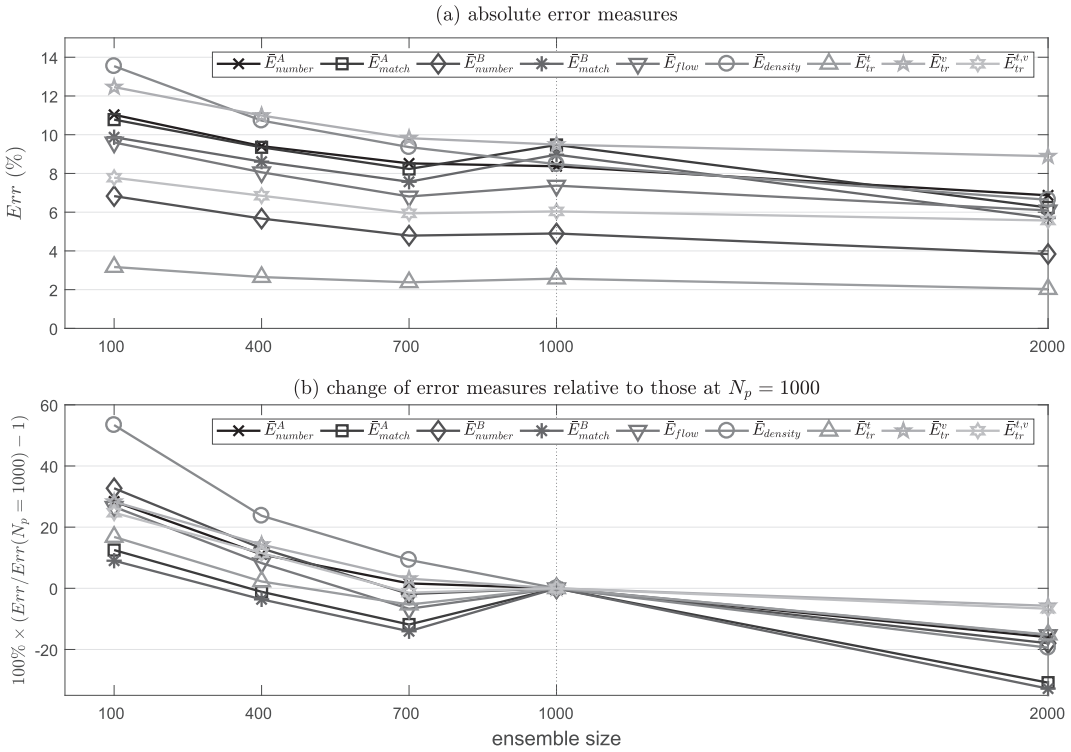


Fig. 14. The influence of N_p on the data assimilation results ($p = 0.9, \lambda = 1/300\ \mathrm{s}^{-1}$).

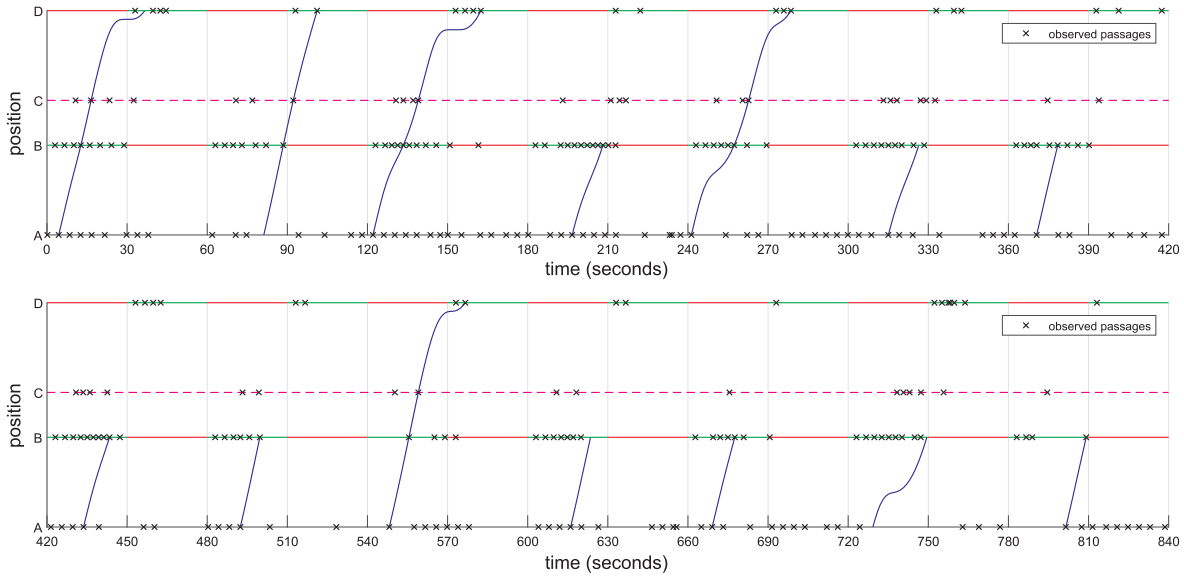


Fig. 15. The sampled trajectories.

6.2. *Incorporating probe vehicle data into the framework*

The proposed data assimilation framework relies on the vehicle accumulation and vehicle passages to reconstruct vehicle trajectories, but it is also open to other types of data if they are available. In this final section, we briefly sketch how, for example, we can assimilate probe vehicle data (sampled vehicle trajectories) in the framework. To demonstrate this we consider the case where one such a sampled vehicle trajectory in each cycle is available, see Fig. 15. For each sampled trajectory, we assume we have available its position every second. Accordingly, we need to adjust the weight computation presented in Section 4.4 by adding a process in which we consider three facts: (1) is there a trajectory in the particle which has a *similar* entering time (i.e. passing time at sensor A) with that of the sampled trajectory? By 'similar', we mean that the difference between the two entering times is smaller than the minimum time headway; (2) if the first condition is met, we check if the corresponding trajectory in the particle passes the same sequence of sensors as the sampled trajectory does; (3) if the first two conditions are both met, we finally look at their average position difference over all sampled time instants. Combining the three facts (differences), we assign a reasonable weight for each particle; since the vehicle passages and the probe vehicle data are independent with each other, the final weight is updated by simply multiplying the weight computed in Section 4.4 by the weight obtained from the probe vehicle data.

The results are shown in Table 11. Clearly, probe vehicle data improves the performance in terms of most metrics, and expectedly, the most significant improvements relate to the translation & distortion errors, and number of reconstructed vehicle passages. Sampled trajectories are direct samples of what happens between two locations, the accuracy of individual vehicle trajectory reconstruction can thus be improved; furthermore, sampled trajectories contain information of passing times by any locations, which can help to detect miss-counts and remove over-counts, therefore we can reconstruct more accurate number of vehicle passages.

Table 11
The data assimilation results when sampled trajectories are available ($p = 0.9, \lambda = 1/300\ \mathrm{s}^{-1}, N_p = 1000$). In each table cell the median error over the 10 simulations is shown along with (in brackets underneath) the 25^{th} and 75^{th} percentiles.

Assimilate sampled trajectories?	Reconstructed vehicle passages				Generalized flow and density		Translation/distortion errors		
	$\bar{E}^{A}_{number}$ (%)	$\bar{E}^{A}_{match}$ (%)	$\bar{E}^{B}_{number}$ (%)	$\bar{E}^{B}_{match}$ (%)	$\bar{E}_{flow}$ (%)	$\bar{E}_{density}$ (%)	$\bar{E}^{t}_{tr}$ (%)	$\bar{E}^{v}_{tr}$ (%)	$\bar{E}^{t,v}_{tr}$ (%)
No	9.37 (7.70, 11.18)	10.47 (7.81, 11.02)	5.90 (5.09, 6.64)	9.97 (7.15, 10.18)	8.37 (7.17, 9.65)	9.48 (7.64, 11.84)	3.57 (3.14, 3.73)	10.49 (10.16, 11.05)	7.04 (6.77, 7.37)
Yes	7.33 (7.12, 8.91)	8.43 (8.03, 8.94)	4.45 (3.17, 5.83)	7.18 (6.92, 8.36)	6.90 (5.88, 8.48)	8.25 (6.73, 9.56)	2.79 (2.49, 2.91)	9.11 (8.39, 10.02)	6.08 (5.44, 6.42)

7. Conclusions and outlook

In this paper, we proposed a generic data assimilation framework to reconstruct plausible trajectories of vehicles on signalized urban arterials, and as a result also infer relevant macroscopic variables (flow, density). The framework uses a particle filter to assimilate noisy data into a microscopic traffic simulation model. These data include vehicle passage times, traffic signal timings, and coarsely available travel time observations. Particularly errors in vehicle counts are notoriously difficult to address and when ignored or "untreated" these yield unbounded errors in vehicle accumulation (density), which make many methods from literature difficult to apply. Our framework *does* systematically address all types of observation errors and is generic in the sense that any (ensemble of) microscopic simulation models (that implement car following, lane changing, crossing, etc.) can be used. We have tested this framework on a simple simulated case that allowed us to analyze in detail the characteristics of the method and its sensitivity to a number of important parameters that relate to the errors in the data and in the simulation model used.

7.1. Conclusions

Our overall conclusion is that the framework is indeed able to reconstruct plausible vehicle trajectories under realistic error assumptions yielding good performance on both macro- and microscopic error measures. The overall absolute percent errors on reconstructing passage counts are around 7%; whereas reconstructing the departure sequences over stop lines (e.g., getting the actual timestamps correct) is done around a 5% error. Given a 10% probability of miss- and over-counts for all sensors this is a very promising result—recall that without assimilation, these errors are unbounded! The errors in density and flow (using Edie's definitions) are around 5% as well. Also the quality of the reconstructed trajectories in terms of matching locations and speeds was satisfactory, although here we observe a (small) sample of large(r) errors. These, however, are a direct consequence of errors in passage counts and departure sequence, causing a few non-existing queues and thus large trajectory errors.

Sensitivity analysis showed that the framework is quite robust to error assumptions. The (macroscopic) estimation errors are consistently about half the percentages of miss- and over-counts. Even with 40% miss-counts or one over-count every two minutes, all estimation errors are (well) within a 20% range. Similarly, the method is robust to model errors (i.e. differences between the ground-truth driving process and the models used in the framework), although we cannot claim to have tested this exhaustively. Both applying different parameter settings (compared to the car following models generating the ground-truth) or a different car following model altogether does not more than marginally affect the results. This result does, however, emphasize an important underlying point. Clearly, unless we have actual evidence (data), either macroscopically in the form of queue dynamics, or (even better) microscopically in the form of sample trajectories, the quality of the reconstruction largely depends on the validity of the microscopic models used in the framework for the specific case at hand. In our case, this validity is evident, since the ground-truth trajectories are produced by a similar model. In real life, when also lateral and crossing behavior enter the equation, it stands to reason that the resulting trajectories will be farther away from the ground truth.

It turns out that such additional data can be easily incorporated in this framework. We show that probe vehicle data (providing sampled trajectories) indeed improve the estimation results in terms of all micro- and macroscopic error measures.

7.2. Assumptions and limitations

We illustrated the working and characteristics of the particle filter based vehicle trajectory reconstruction method with extensive simulation studies and sensitivity analysis. However, in doing so we necessarily made many assumptions, and by far did not cover all aspects needed to make this approach suitable for large scale implementation.

First of all, in this paper we vary with sensor and model error characteristics under the assumption of ideal sensor placement, i.e., we assume that sensors are deployed at both the inflow boundary and the advanced stop line. This assumption will not likely hold in real-life applications, e.g., some sensors may be placed such that not all vehicles are counted, and some roadstretches may not have both inflow and stop line sensors installed. We can, however, formulate some informed hypotheses on the effects of missing or inaccurate boundary measurements on the basis of the many analyses in this paper and some common (traffic flow theory) sense. In the case of systematically biased inflow measurements (due to placement problems or others) the correction method actually still performs well, given travel time measurement sampling rates are not too long (van Lint and Hoogendoorn, 2015). The reason is that the method is assumption-free with respect to counting errors, which may come from any distribution (zero mean or others). As shown in Section 6.1.2, larger travel time sampling rates deteriorate vehicle accumulation accuracy and as a result trajectory reconstruction accuracy. Generally speaking, more bias in flow measurements can be counter-effected by other complementary data (evidence). We do suspect, however, that with very large bias (say an underestimation of 30% or more), the amount of additional evidence needed will increase sharply. Clearly, without inflow measurements altogether, we have to resort to other methods for vehicle accumulation estimation, based on whatever data is available. The literature offers some directions, and we demonstrated in Section 6.2 that incorporating floating car data may counter-effect this problem at least partially. We show that our method degrades gracefully under adverse conditions (increasing sensor errors, model errors), but there are limits beyond which the problem becomes too underdetermined. The bottom line is that there is a minimum set of (minimum quality) sensor data required to reconstruct plausible vehicle trajectories using micro-simulation. An extensive investigation is needed to determine what these minimum requirements are under a range of different circumstances (topology, control, etc.).

Secondly, to illustrate our rather elaborate data assimilation rationale, we study a small-scale network only, in which we permitted longitudinal driving behavior only. Clearly, we would very much like to extend the method to larger scale urban networks in

which we incorporate all aspects of operational driving. This brings about two large challenges: a methodological challenge and a computational challenge. The methodological challenge relates to the huge increase in the number of degrees of freedom in large traffic networks that need to be estimated in order to use microscopic models in the data assimilation. Think of origin destination demand patterns, route choice patterns, multi-lane queuing, complex intersection control, etc. As mentioned above, the microscopic data assimilation framework we propose in this paper requires a sufficient coverage of sensors to estimate these degrees of freedom within reasonable bounds of certainty, so that plausible trajectories can be reconstructed. The more underdetermined and potentially wrong we initialize the simulation models (particles), the more wrong the reconstructed trajectories will be. Next to this methodological challenge, the computational burden will increase tremendously since the required number of particles will increase exponentially (see Section 3.4) to ensure good coverage of the increased size of the state space. In this case, a parallel and distributed version of our framework needs to be developed to deal with large urban networks, and this is certainly possible (e.g., Marinică et al., 2013). The network topology can be a very useful heuristic to help partition the large urban network to achieve load balancing.

Third, in the sensitivity analysis, we adopted the OAT approach, i.e., we change one-factor-at-a-time, to see what effect this produces on the output. The OAT approach is very simple, and has the benefits that first, any change observed in the output will unambiguously be due to the single variable changed; second, the results can be easily compared, since when we change one variable at a time, we can keep all other variables fixed to their baseline values. Therefore, the OAT approach has been used in many data assimilation research, e.g., Xue et al. (2012), Wang and Hu (2015). However, the OAT approach does not fully explore the input space, since it does not take into account the simultaneous variation of input variables, and more importantly, the nonlinear effects that occur due to the interaction of these input variables. As a result, the OAT approach may hide an even larger variability of the error measures with the inputs/parameters.

7.3. Future research directions

The discussion above gives rise to many future research efforts. These relate to application over larger and more complex traffic networks with more variations in sensor availability and quality, and in driving behavior itself (e.g. lateral behavior and crossing). They also relate to a more thorough sensitivity analysis (rather than the OAT approach we used) over the entire input space using e.g. Monte Carlo techniques. Accordingly, the following directions will be explored in future studies. First and foremost, we need to find an appropriate real-life testbed (which includes ground-truth observations!) and apply the method. Clearly, we (then) need to look at incorporating more realistic and more complex behavior (i.e. full fletched simulation models in the framework, instead of just car following models) and also more realistic sensor deployment schema. This makes sense only if—at the same time—we can assimilate richer (e.g., probe vehicle) data to “cover” these extra degrees of freedom. Second, and related, a particular concern (then) is computational complexity. We tested the sensitivity of the method with respect to the amount of particles (i.e. ensemble size of parallel running simulation models). It turned out that our initial choice of 1000 particles was very conservative; with a (much) smaller ensemble an acceptable deterioration of performance would have been achieved. However, with increased complexity of behavior modeled on a much larger urban network, the number of particles required will likely increase exponentially. Therefore, a parallel and distributed version of our framework should be developed. We certainly believe this is possible. Alternatively, one may look into more efficient sampling or assimilation methods (e.g. ensemble Kalman filtering). Third, more advanced sensitivity analysis approach (e.g. Monte Carlo techniques) needs to be utilized to fully analyze the characteristics of the proposed data assimilation framework by exploring the entire joint inputs/parameters space. Additionally, given information on route or destination choice is available, the method could also be extended to estimate unobserved (destination-specific) turn fractions and/or route choices, the quality of which would be subject of course to the route choice models implemented in the simulation model.

As a final note, the objective of this framework is to reconstruct plausible vehicle trajectories on the basis of the available data. As noted, these trajectories—although now consistent with the measurements—are as good as the validity of the underlying models applied. Arguably the most important line of research for our community therefore is the development of simulation models which can make more valid predictions for urban traffic.

Acknowledgment

The research leading to these results is sponsored by Chinese Scholarship Council (No. 201306110027).

Appendix A. Particle filtering in open urban traffic systems

Suppose vehicles can enter the urban traffic system at a set of inflow boundaries, i.e. *Inflow*; the number of vehicles that have entered the system until time step $k \in \{1,2,...\}$ can be defined as

$$Q_k = N_0 + \sum_{j \in Inflow} \int_0^{k\Delta T} q_j(s)ds$$

where N_0 is the number of vehicles in the system in the beginning, $q_j(s)$ is the flow at the boundary $j \in Inflow$, and ΔT is the measurement interval. The state of all vehicles (including those that have left the study area) can be defined as $S_{k,Q_k} = \{s_k^i\}_{i=1}^{Q_k}$ (the same with Eq. (8), but to highlight its dimension, we put Q_k in its subscript), which can be further separated as

$$S_{k,Q_k} = \{s_k^i\}_{i=1}^{Q_k} = \{s_k^i\}_{i=1}^{Q_k-N_k} \cup \{s_k^i\}_{i=Q_k-N_k+1}^{Q_k}, \quad \text{where}$$
$$N_k = N_0 + \sum_{j\in Inflow} \int_0^{k\Delta T} q_j(s)ds - \sum_{j\in Outflow} \int_0^{k\Delta T} q_j(s)ds \tag{A.1}$$

in which *Outflow* is a set of boundaries where vehicles leave the system. $\{s_k^i\}_{i=1}^{Q_k-N_k}$ represents the states of vehicles that have left the system. Once a vehicle leaves the system, we stop updating its state in the simulation. N_k is actually the number of (active) vehicles in the system at time k. Since the arrivals of vehicles usually obey certain stochastic process (e.g., Poisson process), Q_k is a random variable. $S_{0,Q_0:k,Q_k}$ thus captures the vehicle trajectories on the time–space region $[0,k\Delta T] \times [x_1,x_2]$, where $[x_1,x_2]$ is the study area in the traffic simulation.

The objective of particle filtering is to approximate $p(S_{0,Q_0:k,Q_k}|m_{1:k})$ given all measurements till time k. Based on Bayes' theorem, we have

$$p(S_{0,Q_0:k,Q_k}|m_{1:k}) = \frac{p(S_{0,Q_0:k,Q_k})p(m_{1:k}|S_{0,Q_0:k,Q_k})}{p(m_{1:k})}$$

A recursive update equation can thus be obtained:

$$p(S_{0,Q_0:k,Q_k}|m_{1:k}) = p(S_{0,Q_0:k-1,Q_{k-1}}|m_{1:k-1}) \times \frac{p(m_k|S_{k,Q_k})p(S_{k,Q_k}|S_{k-1,Q_{k-1}})}{p(m_k|m_{1:k-1})}$$

Since Q_k is a random variable, the dimension of $S_{0,Q_0:k,Q_k}$ is also random. Therefore, we cannot use the standard sequential importance sampling algorithm to update the posterior distribution $p(S_{0,Q_0:k,Q_k}|m_{1:k})$ due to the variable dimension of $S_{0,Q_0:k,Q_k}$. This variable dimension problem is addressed in a similar way with that in Godsill et al. (2007), i.e., we first make $S_{0,Q_0:k,Q_k}$ to have a fixed dimension with certain extensions, and then we prove that the extensions have no influence on the weight update. To this end, we define a state with fixed dimension as follows

$$S_{k,K} = S_{k,Q_k} \cup V_{k,K-Q_k}$$

where S_{k,Q_k} is defined in Eq. (A.1); $V_{k,K-Q_k} = \{s_k^j\}_{j=Q_k+1}^{K}$ defines a set of *virtual* vehicles that make $S_{k,K}$ to have fixed dimension K, which is a sufficient large constant such that $K \geqslant Q_k, k = 0,1,2,\ldots,span/\Delta T$, where *span* is the length of the simulation study period. These virtual vehicles have no influence on the evolution of S_{k,Q_k}. Since $S_{k,K}$ has a fixed dimension, the state trajectory $S_{0,K:k,K}$ will have a fixed dimension, and as a result, we can use the sequential importance sampling algorithm to update $p(S_{0,K:k,K}|m_{1:k})$, which is first factorized as

$$p(S_{0,K:k,K}|m_{1:k}) = p(S_{0,Q_0:k,Q_k}|m_{1:k})\pi(V_{0,K-Q_0:k,K-Q_k}|S_{0,Q_0:k,Q_k})$$

where the conditional distribution π complements the state trajectory with variable dimension to a fixed dimension. π can be chosen arbitrarily.

Suppose we have a set of random samples χ_{k-1} to approximate $p(S_{0,Q_0:k-1,Q_{k-1}}|m_{1:k-1})$:

$$\chi_{k-1} = \left\{S_{0,Q_0:k-1,Q_{k-1}}^i, w_{k-1}^i\right\}_{i=1}^{N_p} \tag{A.2}$$

We then update the samples based on certain importance density $q(\cdot)$. We can assume that $q(\cdot)$ can be factorized as

$$q(S_{0,Q_0:k,Q_k}|m_{1:k}) = q(S_{k,Q_k}|S_{0,Q_0:k-1,Q_{k-1}},m_{1:k})q(S_{0,Q_0:k-1,Q_{k-1}}|m_{1:k-1})$$

Then the particles in Eq. (A.2) are updated in two steps. First, these particles are updated to time k by sampling according to

$$q(S_{k,Q_k}|S_{0,Q_0:k-1,Q_{k-1}},m_{1:k})$$

Second, the updated particles are complemented to a fixed dimension by drawing samples from

$$q(V_{0,K-Q_0:k,K-Q_k}|S_{0,Q_0:k,Q_k}) = \pi(V_{0,K-Q_0:k,K-Q_k}|S_{0,Q_0:k,Q_k})$$

The weights in Eq. (A.2) are updated as follows:

$$\begin{aligned} w_k &= \frac{p(S_{0,K:k,K} \mid m_{1:k})}{q(S_{0,K:k,K} \mid m_{1:k})} \\ &= \frac{p(m_k \mid S_{k,Q_k})p(S_{k,Q_k} \mid S_{k-1,Q_{k-1}})p(S_{0,Q_0:k-1,Q_{k-1}} \mid m_{1:k-1})}{p(m_k \mid m_{1:k-1})q(S_{k,Q_k} \mid S_{0,Q_0:k-1,Q_{k-1}},m_{1:k})q(S_{0,Q_0:k-1,Q_{k-1}} \mid m_{1:k-1})} \frac{\pi(V_{0,K-Q_0:k,K-Q_k} \mid S_{0,Q_0:k,Q_k})}{\pi(V_{0,K-Q_0:k,K-Q_k} \mid S_{0,Q_0:k,Q_k})} \\ &\propto \frac{p(m_k \mid S_{k,Q_k})p(S_{k,Q_k} \mid S_{k-1,Q_{k-1}})}{q(S_{k,Q_k} \mid S_{0,Q_0:k-1,Q_{k-1}},m_{1:k})} \frac{p(S_{0,Q_0:k-1,Q_{k-1}} \mid m_{1:k-1})}{q(S_{0,Q_0:k-1,Q_{k-1}} \mid m_{1:k-1})} \\ &= \frac{p(m_k \mid S_{k,Q_k})p(S_{k,Q_k} \mid S_{k-1,Q_{k-1}})}{q(S_{k,Q_k} \mid S_{0,Q_0:k-1,Q_{k-1}},m_{1:k})} \times w_{k-1} \end{aligned}$$

which is again independent of the states of the virtual vehicles. If the system transition density is chosen as the importance density, the weight computation will be simplified to $w_k = p(m_k|S_{k-1,Q_{k-1}:k,Q_k}) \times w_{k-1}$.

References

Arulampalam, M.S., Maskell, S., Gordon, N., Clapp, T., 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Trans. Signal Process. 50 (2), 174–188.

Bhaskar, A., Tsubota, T., Kieu, L.M., Chung, E., 2014. Urban traffic state estimation: fusing point and zone based data. Transport. Res. Part C: Emerging Technol. 48, 120–142.

Brackstone, M., McDonald, M., 1999. Car-following: a historical review. Transport. Res. Part F: Traffic Psychol. Behav. 2 (4), 181–196.

Coifman, B., 2002. Estimating travel times and vehicle trajectories on freeways using dual loop detectors. Transport. Res. Part A: Policy Practice 36 (4), 351–364.

da Rocha, T.V., Leclercq, L., Montanino, M., Parzani, C., Punzo, V., Ciuffo, B., Villegas, D., 2015. Does traffic-related calibration of car-following models provide accurate estimations of vehicle emissions? Transport. Res. Part D: Transport Environ. 34, 267–280.

Daganzo, C.F., 2005a. A variational formulation of kinematic waves: basic theory and complex boundary conditions. Transport. Res. Part B: Methodol. 39 (2), 187–196.

Daganzo, C.F., 2005b. A variational formulation of kinematic waves: solution methods. Transport. Res. Part B: Methodol. 39 (10), 934–950.

Djurić, P.M., Kotecha, J.H., Zhang, J., Huang, Y., Ghirmai, T., Bugallo, M.F., Miguez, J., 2003. Particle filtering. IEEE Signal Process. Mag. 20 (5), 19–38.

Edie, L.C., 1963. Discussion of traffic stream measurements and definitions. In: Proceedings of the 2nd International Symposium on the Theory of Traffic Flow. Paris, France, pp. 139–154.

Godsill, S., Vermaak, J., Ng, W., Li, J., 2007. Models and algorithms for tracking of maneuvering objects using variable rate particle filters. Proc. IEEE 95 (5), 925–952.

Goodall, N.J., Smith, B.L., Park, B.B., 2016. Microscopic estimation of freeway vehicle positions from the behavior of connected vehicles. J. Intell. Transport. Syst. 20 (1), 45–54.

Kesting, A., Treiber, M., 2008. Calibrating car-following models by using trajectory data: methodological study. Transport. Res. Rec.: J. Transport. Res. Board 2088, 148–156.

Lu, X., Varaiya, P., Horowitz, R., Palen, J., 2008. Faulty loop data analysis/correction and loop fault detection. In: 15th World Congress on Intelligent Transport Systems and ITS America's 2008 Annual Meeting. New York, NY.

Marinică, N.E., Sarlette, A., Boel, R.K., 2013. Distributed particle filter for urban traffic networks using a platoon-based model. IEEE Trans. Intell. Transport. Syst. 14 (4), 1918–1929.

Mehran, B., Kuwahara, M., Naznin, F., 2012. Implementing kinematic wave theory to reconstruct vehicle trajectories from fixed and probe sensor data. Transport. Res. Part C: Emerging Technol. 20 (1), 144–163.

Montanino, M., Punzo, V., 2015. Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns. Transport. Res. Part B: Methodol. 80, 82–106.

Nantes, A., Ngoduy, D., Bhaskar, A., Miska, M., Chung, E., 2016. Real-time traffic state estimation in urban corridors from heterogeneous data. Transport. Res. Part C: Emerging Technol. 66, 99–118.

Punzo, V., Montanino, M., 2016. Speed or spacing? cumulative variables, and convolution of model errors and time in traffic flow models validation and calibration. Transport. Res. Part B: Methodol. 91, 21–33.

Punzo, V., Montanino, M., Ciuffo, B., 2015. Do we really need to calibrate all the parameters? variance-based sensitivity analysis to simplify microscopic traffic flow models. IEEE Trans. Intell. Transport. Syst. 16 (1), 184–193.

Saifuzzaman, M., Zheng, Z., 2014. Incorporating human-factors in car-following models: a review of recent developments and research needs. Transport. Res. Part C: Emerging Technol. 48, 379–403.

Sun, Z., Ban, X.J., 2013. Vehicle trajectory reconstruction for signalized intersections using mobile traffic sensors. Transport. Res. Part C: Emerging Technol. 36, 268–283.

Sun, Z., Hao, P., Ban, X.J., Yang, D., 2015. Trajectory-based vehicle energy/emissions estimation for signalized arterials using mobile sensing data. Transport. Res. Part D: Transport Environ. 34, 27–40.

Treiber, M., Helbing, D., 2002. Reconstructing the spatio-temporal traffic dynamics from stationary detector data. Cooperative Transport. Dynam. 01, 3.1–3.24.

Treiber, M., Hennecke, A., Helbing, D., 2000. Congested traffic states in empirical observations and microscopic simulations. Phys. Rev. E 62 (2), 1805–1824.

Treiber, M., Kesting, A., 2013. Traffic Flow Dynamics: Data, Models and Simulation. Springer, Berlin Heidelberg.

van Hinsbergen, C.P.I.J., Schreiter, T., Zuurbier, F.S., van Lint, J.W.C., van Zuylen, H.J., 2012. Localized extended Kalman filter for scalable real-time traffic state estimation. IEEE Trans. Intell. Transport. Syst. 13 (1), 385–394.

van Leeuwen, P.J., 2009. Particle filtering in geophysical systems. Mon. Weather Rev. 137 (12), 4089–4114.

van Lint, J.W.C., 2010. Empirical evaluation of new robust travel time estimation algorithms. Transp. Res. Rec. 2160, 50–59.

van Lint, J.W.C., Hoogendoorn, S.P., 2010. A robust and efficient method for fusing heterogeneous data from traffic sensors on freeways. Comput.-Aided Civ. Infrastruct. Eng. 25 (8), 596–612.

van Lint, J.W.C., Hoogendoorn, S.P., 2015. A generic methodology to estimate vehicle accumulation on urban arterials by fusing vehicle counts and travel times. In: Transportation Research Board 94th Annual Meeting. No. 15-5134.

van Wageningen-Kessels, F., van Lint, J.W.C., Vuik, K., Hoogendoorn, S.P., 2015. Genealogy of traffic flow models. EURO J. Transport. Logist. 4 (4), 445–473.

Wang, M., Hu, X., 2015. Data assimilation in agent based simulation of smart environments using particle filters. Simul. Model. Pract. Theory 56, 36–54.

Wang, Y., Papageorgiou, M., Messmer, A., 2006. RENAISSANCE – A unified macroscopic model-based approach to real-time freeway network traffic surveillance. Transport. Res. Part C: Emerging Technol. 14 (3), 190–212.

Xue, H., Gu, F., Hu, X., 2012. Data assimilation using sequential Monte Carlo methods in wildfire spread simulation. ACM Trans. Model. Comput. Simul. 22 (4), 23:1–23:25.

Yuan, Y., van Lint, J.W.C., Wilson, R.E., van Wageningen-Kessels, F., Hoogendoorn, S.P., 2012. Real-time Lagrangian traffic state estimator for freeways. IEEE Trans. Intell. Transport. Syst. 13 (1), 59–70.