Particle filter state estimator for large urban networks

Nicolae Marinică and René Boel

***Abstract*— This paper applies a particle filter (PF) state estimator to urban traffic networks. The traffic network consists of signalized intersections, the roads that link these intersections, and sensors that detect the passage time of vehicles. The traffic state $X(t)$ specifies at each time time t the state of the traffic lights, the queue sizes at the intersections, and the location and size of all the platoons of vehicles inside the system. The basic entity of our model is a platoon of vehicles that travel close together at approximately the same speed. This leads to a discrete event simulation model that is much faster than microscopic models representing individual vehicles. Hence it is possible to execute many random simulation runs in parallel. A particle filter (PF) assigns weights to each of these simulation runs, according to how well they explain the observed sensor signals. The PF thus generates estimates at each time t of the location of the platoons, and more importantly the queue size at each intersection. These estimates can be used for controlling the optimal switching times of the traffic light.**

***Index Terms*— Bayesian estimation, particle filtering, urban traffic, platoon based model, stochastic systems.**

I. Introduction

Estimation and prediction of the traffic state in an urban network is an important component of the feedback loop used in on-line road traffic management. Both PF and the distributed model predictive controllers (MPC) that we use for adapting the switching times, require a fast and simple model describing the location of vehicles in the network. Many different models for both freeway traffic and for urban traffic have been developed. Urban traffic, the topic of this paper, can be described by macroscopic models [4], [5] representing the average traffic behavior in terms of the aggregated variables density and flow, as measured at different locations, or by microscopic models that represent the behavior of each vehicle individually. Microscopic models are suitable for very low density traffic, where no feedback control is necessary, while macroscopic models are suitable for oversaturated traffic where control actions select cycle times, red/green split, and offset.

Our work deals with the intermediate traffic load case, and we aim at selecting the actual switching times of the individual traffic lights using local feedback, but trying to maintain as much as possible the green wave, so as to postpone the onset of saturation as much as possible. This requires PF traffic state estimators and MPC controllers that depend on a fast simulations models that can approximately locate all vehicles.

In order to efficiently achieve this goal our model groups vehicles into platoons. A platoon consists of vehicles that travel at approximately the same speed, closely following each other. The state $X(t)$ of the traffic network at time t in such a platoon based model represents the status of each traffic light, the queue sizes at each approach lane of each intersection, and the location of the head, and the size, of each platoon inside the network. At various locations in the network (e.g. just upstream and just downstream of each intersection) sensors detect the successive passage times of vehicles. These measurements, together with information on the red/green switching times of all the traffic lights, are used to recursively estimate the size of the queues, and the location and size of the platoons.

Note that the problem we are posing is simpler than the problem dealt with in [19] where the sensors provide data only on the fraction of time a loop detector is covered by a vehicle. Vigos et al. in [19] must transform their time-occupancy data into space-occupancy data. This problem becomes difficult for urban traffic because the intersections, and especially the traffic lights, cause very strong inhomogeneity in the traffic flow. The platoon based model used in this paper represents this inhomogeneity explicitly, thus simplifying the estimation problem.

We assume that a link has at least one sensor at its upstream and at its downstream edge, and that these sensors detect, with some noise, the passage times of vehicles. It seems in principle trivial to estimate the current number of vehicles in this link (vehicles that are either part of a platoon moving through the link, or that are stopped in a queue if the outflow of the link is blocked), simply by subtracting outflow count from inflow count. This is impossible not only because the initial number of vehicles in the link is unknown, but also because the sensors are very unreliable (for some sensors we believe the rate of missed detections can be up to 30%). The error of such a naive estimate would be a random walk with linearly increasing variance, due to the noise in the sensors. Fortunately combining the sensor data with the dynamical model (e.g. the conservation laws relating flows at successive sensors along a link) allows reduction of the errors in the estimated states. For example the model allows the estimator to deduce that the queue size is 0 at an intersection if the outflow sensor for a flow that has green light sees no traffic. The first platoon leaving an intersection after a traffic light switches from red to green gives information on the size of the queue just before this red/green switch.

In [19] recursive state estimation is achieved via a Kalman filter combining sensor data with the dynamical traffic model, esp. the conservation equations. For the macroscopic METANET freeway model [7] an extended Kalman filter has been used by e.g.[6] [12], but this requires a linearization something that is not feasible for our discrete event platoon

Nicolae Marinică is with the SYSTeMS Research Group, Faculty of Engineering, University of Ghent, B-9052 Zwijnaarde, Belgium nicolaeemanuel.marinica@ugent.be

René Boel is with the SYSTeMS Research Group, Faculty of Engineering, University of Ghent, B-9052 Zwijnaarde, Belgium Rene.Boel@UGent.be

based model, expressing the traffic inhomogeneity. We need general Bayesian recursive filtering algorithms, but unfortunately their computational complexity is prohibitive for large systems. The recursive calculation of conditional densities of the current state given the measurements requires integration over the state space, which is too large for the platoon based model.

Recently particle filtering (PF) has been proposed as a method for approximating the conditional density by an empirical histogram obtained via Monte Carlo simulation. There is already a rich literature about PF for traffic estimation for freeway traffic [1], [2], [15] and [3], but little about state estimation for urban traffic. In this paper we show that this PF approach can also be used for estimating the state of a platoon based model of the urban traffic. This requires the generation of N random simulation runs of the platoon based model of the urban traffic. The platoon based model used in this paper can indeed be implemented by an efficient discrete event system (DES) simulator, because it needs to keep track only in its event list of the arrival times and size of the successive platoons at a few locations (e.g. upstream of an intersection, sensor locations). Since the number of events to be executed by the simulator is proportional to the number of platoons, much smaller than the number of vehicles, this significantly reduces the simulation time.

Our choice for a platoon based model is motivated by the fact that it leads to fast simulation, and by the fact that platoon arrival times at intersections are used in our feedback control algorithm for selecting the switching times of traffic lights. Moreover the traffic data that we have available, covering measurements over more than 40 days in a network with 5 signalized intersections, 2 roundabouts, and a number of unsignalized intersections, indicate that vehicles travel most of the time in platoons. Moreover the data show that (for the day time traffic that we want to control) the number of platoons per time unit is fairly constant: the expected size of the platoons grows with the traffic intensity while the arrival rate of the platoons remains approximately constant. Figure 1 shows the variation in the number of platoons (for each working day during 1 week) at a location on a road sufficiently far from the exit of a roundabout and $250[m]$ upstream from an intersection, as obtained from the data set.

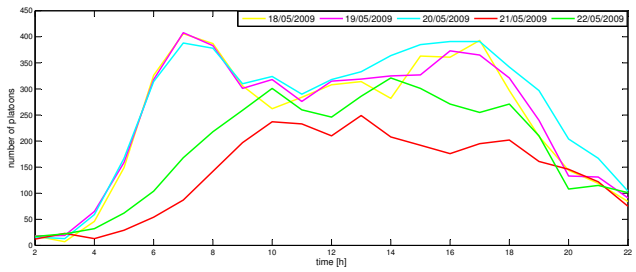

Fig. 1. Number of platoons (defined with inter-vehicle time 5 sec) for 5 different working days.

The platoon based model is presented in Section II-A. The stochastic simulation set-up of the particles is tackled in Section II-B. Section III explains the PF implementation for a platoon based model and some ideas about the distributed implementation of the algorithm for large networks. Section IV demonstrates the feasibility and the robustness of our estimators on some simple scenarios, while Section V provides some conclusions, and discusses some further improvements.

II. The model

A. Discrete event systems model

This section briefly describes a discrete event model of the dynamic behavior of urban traffic, also introduced in [18]. Vehicles that are closer to each other than a certain inter-platoon distance $\Delta(=6[sec]$ in this paper) are assumed to be part of one single platoon of vehicles. The time distance $I_{interveh,k}$ between vehicles k and $k+1$ in the same platoon is a random variable taking values in an interval $[d_{min},6[sec])$. The vehicles in these platoons move, as time t evolves, from their origin, where they are generated by a *source* component, to their destination, as they pass through the successive *link* and *intersection* components of the urban traffic network $\mathscr{U}$. Platoons slow down, merge, and split up by their interaction with each other and with the traffic lights at signalized intersections. Looking at this space-time description of platoons at a given time t defines the *state* X_t of the model:

- location and size of all the platoons in each link of $\mathscr{U}$, and size of the queues at each intersections, at time t;
- red/green/yellow mode of all traffic lights, at time t.

This state can be defined compositionally as a stack $X(t) = (X_c(t), c \in \mathscr{L} \bigcup \mathscr{I}, M_i(t))^T$ of substates. Here $\mathscr{L}$ is the set of all links in $\mathscr{U}$, while $\mathscr{I}$ is the set of all intersections. For $Link_\ell \in \mathscr{L}$ the substate $X_\ell(t)$ represents the position and size of each platoon in $Link_\ell$, while for intersection $Int_i \in \mathscr{I}, X_i(t)$ represents the queue sizes at all its pre-selection lanes; $M_i(t)$ represents the state of a timed automaton expressing the behavior of the traffic lights (current phase of the traffic light, but also values of clocks that restrict future switching times) at Int_i.

Figure 2 represents a small part of an urban traffic network, with a $Link_i$ of length L_i that is connected at its upstream access point to the intersection C_1. The downstream exit point of $Link_i$ is connected to the 3 pre-selection lanes (traffic going straight, turning right, or turning left) of intersection C_2. The nodes of a general urban traffic network $\mathscr{U}$ connect the exit point of one component to the access point of another component. A sensor is installed at each node in $\mathscr{U}$ detecting the passage of platoons by measuring the time at which the lead vehicle of the successive platoons pass the sensor location, and by also measuring the size of the platoon. In other words the state $X(t)$ represents a vertical cut at time t through the space-time diagram [20] of the platoon evolution, while a sensor measures a horizontal cut, at the sensor location, through this space-time diagram.

Remark: Our definition of a platoon is a gross simplification of real system behavior. Indeed the appropriate inter-platoon distance Δ depends on the speed of the vehicles. However the data we have available only show the times at

which vehicles pass a sensor location, not their speed. We can hence not distinguish whether a large time delay between 2 successive vehicles implies they belong to different platoons, or whether they belong to the same platoon driving so slowly that their time distance is large. For urban traffic it is reasonable to assume that most of the time the vehicle speed is either normal (between 80% and 100% of the maximal speed), or else is very low (even equal to 0), which of course makes our definition of a platoon meaningless at an intersection with a red light. Therefore we extend the definition of a platoon by designating a queue of stopped vehicles as a (stationary) platoon. These queues behind a red light can be recognized as the first platoon that passes the sensor at the access point of the downstream link, as soon as the traffic light turns green. Our approximations agree with the data we have available, and moreover provide us with the variables that we will use later on in feedback control laws of traffic lights.

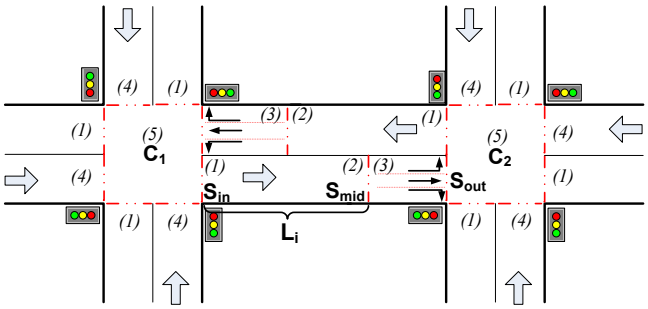

Fig. 2. The road topology and the sensors

Our platoon based model defines how the platoons evolve inside each component, and how they are moving from an upstream component to a downstream component. We define below the different types of components of an urban traffic network, specifying how the state of the platoons evolves inside the component (in particular how the events are defined that describe the evolution as a D.E.S.), and how one component is connected to another component:

• Traffic $Source_r \in \mathscr{R}, r = 1,\ldots,R = \sharp\mathscr{R}$ generates at successive times $T_{r,n,head}, n = 1,\ldots$ platoons of size $K_{r,n}$ entering the upstream access point of $Link_\ell$ (at the edge of the traffic network). The size $K_{r,n}$ is a random variable with time dependent (empirically determined) probability distribution $\wp(K_{r,n} = k) = P_r(k,t), k = 1,2,\ldots,K_{max}(t)$. The duration $D_{r,n} = T_{r,n,tail} - T_{r,n,head}$ of this n-th platoon, that is the difference between the arrival time $T_{r,n,tail}$ at the source of the last vehicle of the platoon and the arrival time $T_{r,n,head}$ of its first vehicle, is the sum of $K_{r,n}$ random variables $I_{nterveh,k}$ with mean $EI_{nterveh}$. The time gap $D_{r,n} = T_{r,n+1,head} - T_{r,n,tail}$ between successive platoons is random, defined by $D_{r,n} = 6[sec] + G_{r,n}$, where $G_{r,n}$ are independent random variables with exponential distribution, and mean value EG_r. This results in a time varying arrival rate of $\lambda(r,t) = \frac{EK_r(t)}{6+EG_r+EK_r(t).EI_{interveh}}[veh/sec]$. The parameters of this model must be determined based on historical measurements at the location of $Source_r$.

• The head of a platoon that enters $Link_\ell, \ell = 1,\ldots,L = \sharp\mathscr{L}$ at its access point at time $T_{n,\ell}$ ($= T_{r,n,head}$ if this access point is connected to $Source_r$) reaches the exit point of $Link_\ell$ at time $T_{n,\ell} + L_\ell/v_{n,\ell}$ with a random delay, depending on the length L_ℓ of $Link_\ell$ and on the random speed $v_{n,\ell} = p.V_{max}$ where $p = 1, 0.9$, or 0.8 with probabilities $0.8, 0.15, 0.05$ in our current implementation (accidents can be modeled by large random increase in this random delay). $Link_\ell$ has a predefined storage capacity (in order to model spill-backs). The size of the platoon that enters the $Link_\ell$ is inherited from the upstream component, e.g. if the upstream component is $Source_r$ generating a platoon of size $K_{r,n}$ then the size of this platoon remains $K_{r,n}$. In our current implementation we consider merging of platoons only at the end of $Link_\ell$, if the head of the faster platoon n catches up with the tail of the slower platoon $n-1$, (including the case where a queue is formed at the blocked exit of the link). In order to limit the size of the paper we omit details on randomly splitting up platoons in a link.

• An *intersection* $Int_i, i = 1,\ldots,I$ has as many access points as there are pre-selection lanes for traffic arriving at its approach roads, and has as many exit points as there are downstream exit roads. If a platoon of size $K_{r,n}$ reaches the exit point of a link, that is connected to the entrance point of the intersection Int, then the vehicles are assigned to the q pre-selection lanes according to a multinomial distribution with parameters $K_{r,n}, P_q(t)$ where the parameters $P_q(t)$ are determined on the basis of historical data at Int_i. Provided the priority and safety rules (mode of traffic lights, and additional right of way rules for left turning traffic) are satisfied a platoon in a pre-selection lane moves to the appropriate exit point that is the entrance point of a downstream link. Queues form behind a traffic light during its red phase. As soon as the traffic light turns green the queue starts moving and leaves as one newly formed platoon until the queue has become empty. When the queue is empty, during a green phase, platoons move from access point of the intersection to the exit point, with a random delay.

• At least one $Sensor_s \in \mathscr{S}, s = 1,\ldots,S = \sharp\mathscr{S}$ is installed at each node in $\mathscr{U}$, but more sensor locations along a link can be included. One purpose of sensors is to provide the measurement data to be used by the particle filter. However we can also introduce virtual sensors, in order to simplify the modeling. These sensors do not generate measurement data, but they allow a more accurate model. Indeed in the description of a link we assumed that platoons do not split up nor merge inside a link. This is valid as an approximation of real traffic behavior only if the length of the links is sufficiently small. In particular this assumption guarantees that our traffic model is a pure discrete event model, where the evolution of the traffic behavior is completely determined by the event times, when the head or the tail of a platoon arrives at a sensor location (and by the red-green switching times).

B. Discrete event simulation tool

The platoon based model introduced in Section II.A is a DES model specifying the progress of platoons of vehicles traveling through the components of the network. The set of platoons $P_p, p \in PI(t)$ that are present in the network at time t is defined by the index set $PI(t) \subset N_0$. The location and the size of platoon P_p at time t can be calculated from the arrival time of the first vehicle of platoon P_p at the sensor location just upstream of the current position of P_p, and from its size. Since platoons do not change speed nor size in between sensors this information is sufficient for uniquely determining the substate $X_c(t), c \in \mathscr{L} \bigcup \mathscr{I}$. This information is also sufficient to implement the model in a discrete event simulation tool using an agenda that keeps track of all the future events: *platoon-entering intersection; platoon-entering link; platoon-leaving intersection; platoon-leaving link; next-green; platoon-generated-by-source*. This agenda provides the following information for each event: $e = \{time, action\_type, place, size\}$ where *place* represents the set of sensors where the platoons are detected at the nodes of $\mathscr{U}$ (and possibly at other places where the sensors are deployed on a link). The *action_type* represents the type of the event that has to take place at time *time*. The *size* variable is used for different purposes depending on the *action_type* (number of cars that have to move, duration of next green period).

The model of section II.A indeed allows the simulation tool to update the agenda each time an event is executed, since platoons only change speed nor size at the boundary of different components. Knowing the current state $X(T_k)$ just before the time T_k when the k-th event occurs it is possible to calculate the time of the next events to be added to the event list. Arrival times of first and last vehicles of platoon P_p at a any sensor location are determined by the arrival time of first vehicle and size of P_p at the upstream sensor location, and by the time delay along the component in between the 2 sensors (along a link this depends on the randomly selected speed, in an intersection on the random delay at the intersection and for left turning traffic also on higher priority traffic). Switching times for the phase of a traffic light are determined by the feedback control law for the intersection, which is beyond the scope of this paper, but which is easy to implement in the simulator.

III. Technical description of Particle Filter

The measurements provided by the active sensors $sensor_j, j \in J \subseteq \mathscr{S}$ can be used to estimate the current state $X(t)$ (or to predict future state evolution $X(\tau), \tau \geq t$, so that MPC controllers can be implemented). The output at an active $sensor_j, j \in J$ is generated as follows. Each time a vehicle passes the location of $sensor_j$ this generates, with probability $P_{detect,j}$, independently of the other pulses that are generated, a pulse in the output of $sensor_j$. The sensor output thus fails to detect $(1 - P_{detect,j}).100\%$ of the vehicles. The sensor also makes random errors by registering some false detection pulses at times when no vehicle is passing the sensor location.

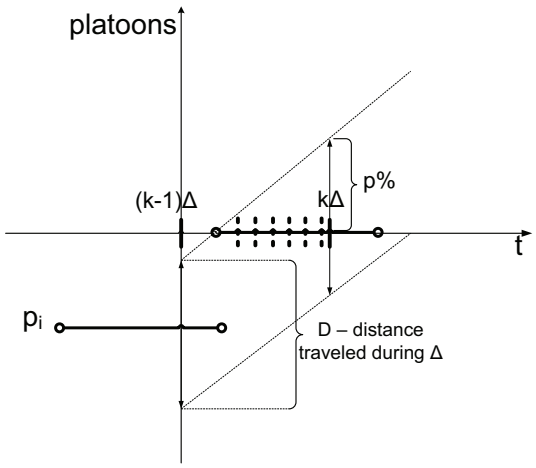

Fig. 3. Sensor S_j detecting a platoon during Δt

The output of $sensor_j, j \in J \subseteq \mathscr{S}$ is sampled per interval of length Δ_{sample}, i.e. at the end of the k-th sampling period $[t_k, t_{k+1}) = [k.\Delta_{sample}, (k+1).\Delta_{sample})$ $sensor_j$ generates a random signal $Y_{sensor_j}(k)$ that depends on the system state $X(\tau), \tau \in [t_k, t_{k+1})$ (the location and size of the platoons in the components upstream and downstream of $sensor_j$ at time t_k), and on the random noise due to missed vehicles and false detections. The number of vehicles that pass the sensor location is determined as shown in Figure 3. For each platoon P_p let ρ_p be the fraction of the interval $[T_{n,j,head}, T_{n,j,tail})$ that overlaps (and intersects) the sampling interval $[t_k, t_{k+1})$. If the sensor worked perfectly it would output $Z_{j,k} = \sum_{p \in PI(t)} \rho_p . K_{n,j}$ (usually Δ_{sample} is smaller than the gap $D_{j,n}$ between platoons so that only one platoon must be considered in this sum). The noisy output $Y_{j,k} = \tilde{Z}_{j,k} + V_{j,k}$ of $sensor_j$ is then the sum of $\tilde{Z}_{j,k}$, a binomial random variable with parameters $Z_{j,k}$ and probability $P_{detect,j}$, and the random number $V_{j,k}$ of false detections (which in the current implementation has a probability distribution $\mathscr{P}(V_{j,k} = n) = p_n$ with $p_0 = 0.95,\ p_1 = 0.04,\ p_2 = 0.01$). Further analysis of experimental data is needed in order to obtain a more accurate distribution of the false detections.

This clearly relates the sensor output signals to the evolution of the state of the D.E.S., since it specifies $\mathscr{P}(Y_{j,k} = m \mid X(t_k))$ (in fact only the substates of $X(t_k)$ that describe the components upstream and downstream of $sensor_j$ appear in the conditioning). Let Y_k denote the vector of all observations $Y_{j,k}, j \in J$ then the assumption that the noises at different sensors are independent makes it easy to write $\mathscr{P}(Y_k \mid X(t_k))$ as a product of the local conditional probabilities. Hence it is in principle possible to carry out the measurement update step of a recursive Bayesian estimation algorithm by applying Bayes' rule: the multiplication $\mathscr{P}(X(t_k) \mid Z^{k-1}).\mathscr{P}(Y_{j,k} = m \mid X(t_k))$, where $Z^k = \{Y_1, Y_2, \ldots, Y_{k-1}, Y_k\} = \{Z^{k-1}, Y_k\}$, followed by a normalization defines the conditional probability distribution (PDF) $\mathscr{P}(X(t_k) \mid Z^k)$ of the current state given all the measurements available up to time t_k. The random D.E.S. model of Section II.A implicitly defines the transition probability $\mathscr{P}(X(t_k) \mid X(t_{k-1}))$. Applying the law of total

probability

$$\mathscr{P}\left(X(t_k)|Z^{k-1}\right) = \int \mathscr{P}\left(X(t_k)|u\right)\mathscr{P}\left(u|Z^{k-1}\right).du \quad (1)$$

(for u ranging over the space of possible values of $X(t_{k-1})$) implements the state update step of the recursive Bayesian estimator.

The integration necessary to evaluate the denominator in the measurement update step - necessary in order to normalize the PDF - is computationally too expensive when the state space is too large, as is the case for the platoon based model of traffic network $\mathscr{U}$. The PF makes the problem scalable by replacing this integration by the average over a fixed number N of random samples of the traffic behavior. These sample paths $Part_{n,\ell}, 0 \leq \ell \leq k$, called *particles*, can be generated using N independent Monte Carlo simulations running in parallel on a computer, each simulations run $i = 1,\ldots,N$ generating a state trajectory $x^i_\ell, \ell = 1,\ldots,k$, implementing the dynamical model of the plant as explained in Section II-B. Generating the particles by random simulation also avoids the problem that while the explicit formula for the transition probability distribution $\mathscr{P}(X(t_k) \mid X(t_{k-1}))$ is very complicated, generating independent random samples of the state evolution in the interval $[k.\Delta_{sample}, (k+1).\Delta_{sample})$is easy by using the D.E.S. simulation tool of Section II.B.

A weight $w^n_k, n = 1,\ldots,N$ is associated to each of the N particles $Part_{n,\ell}, 0 \leq \ell \leq k$. The PF method approximates $\mathscr{P}(X(t_k)|Z^k)$ by the empirical histogram corresponding to the current state $X_{part,n}(t_k))$ of the n-th *particle* $Part_{n,\ell}, 0 \leq \ell \leq k$, and by its weight w^n_k [13] [14]. This interpretation is valid provided that the weight w^n_k expresses reasoably accurately at time t_k how well the particle $Part_{n,\ell}, 0 \leq \ell \leq k$. explains the observations Z^k, that are available at moment t_k. The weight w^n_k should thus be multiplied, at each time t_k when an observation is obtained by the likelihood $\mathscr{P}(Y_{j,k} = m \mid X_{part,n}(t_k))$, imitating the Bayesian measurement update step in the recursive Bayesian estimator (note that when using a PF estimator it is easy to consider the case where the observed output $Y_{j,k}$ depends not just on the most recent state of the system, but also on past behavior, as might be the case in practical implementations for road traffic). In the practical implementation, the sum of the weights must be normalized to 1 for numerical reasons, but this only requires summing N terms, not an integration over a very large state space.

In the case study used in this paper to validate the approach we apply PF in order to estimate the queue size upstream from sensor S_{out} at intersection C_2, and the location of the platoons in *link* L_i, as shown in Figure 2. Note that these are the variables that should be used in order to adapt the traffic lights at C_2 to the arrival stream of traffic coming from intersection C_1. In order to appreciate the difficulty of the presented estimation problem consider also the fact that we only measure traffic flow, which may be 0 either because there are few vehicles ($traffic\ density\ \approx 0$) or because their speed is very low (≈ 0). The queue size essentially integrates the difference in observed inflow, at S_{mid}, and the observed outflow at S_{out}, but the error in this integration behaves like a random walk. The estimate can be improved by taking into account the phase of the traffic light, by observing that very low outflow rates at S_{out} indicate an empty queue, and that immediately after the traffic light turned green one can expect to see a platoon at S_{out} equal in size to the queue at the end of the red phase. The PF estimator efficiently combines all this information.

Algorithm 1 Particle filter algorithm

Ensure: $\rightarrow$ Initialization of the particles $k = 0$
1: **for** $i = 1$ to N **do**
2: generate the i^{th} sample $\{x^i_0\}$
3: initialize i^{th} weight with $w^i_0 = 1/N$
4: **end for**
5: **for all** k such that $k \geq 1$ **do**
Ensure: $\rightarrow$ Prediction step
6: **for** $i = 1$ to N **do**
7: update the i^{th} sample $\{x^i_k\}$ according to $p\left(x_k|x^i_{k-1}\right)$
8: **end for**
Ensure: $\rightarrow$ Measurement step
9: **for** $i = 1$ to N **do**
10: update the weight of the i^{th} sample $w^i_k = w^i_{k-1} \cdot p\left(z_k|x^i_k\right)$
11: **end for**
Ensure: $\rightarrow$ Normalization
12: **for** $i = 1$ to N **do**
13: normalize the weight of the i^{th} sample $\hat{w^i_k} = {w^i_k}/{\sum_{i=1}^N w^i_k}$
14: **end for**
Ensure: $\rightarrow$ Output
15: order particles by the size of the weight.
16: $\hat{x}_s \Leftarrow$ particle with the highest weight.
Ensure: $\rightarrow$ Resampling
17: **for** $i = 1$ to $N/2$ **do**
18: for pair $\left(P^i_k, P^{N-i+1}_k\right)$
19: replace the state of the particleP^{N-i+1}_k with the state of the particle P^i_k
20: $w^{i_new}_k = \left(w^i_k + w^{N-i+1}_k\right)/2$
21: $w^i_k = w^{i_new}_k$ and $w^{N-i+1}_k = w^{i_new}_k$
22: **end for**
Ensure: $\rightarrow$ Time step update
23: $k \Leftarrow k+1$
24: **end for**

The PF algorithm is described in algorithm 1. Initially N random selections of sizes and locations of platoons $Part_{n,0}$ are generated, each with weight $1/N$. From then on one implements for $k = 0, 1, \ldots,$ the following recursion. Generate for each particle $Part_{n,\ell}, \ell \leq k$ the state evolution in the interval $[t_k, t_{k+1})$ according to the model of Section II.B (see line 7 in algorithm 1). At time t_k, when the information Z^k is received, the algorithm updates the weight w^n_k of the n-th particle by multiplying it with $\mathscr{P}\left(Y_k|x^n_k\right)$ (see line 10 in algorithm 1). For large N the particles $Part_{n,\ell}, \ell \leq k, n = 1,\ldots,N$ will generate an empirical histogram that approximates the true conditional PDF $\mathscr{P}(X(t_k) \mid Z^k)$ sufficiently accurately.

Because the parallel simulation of the N particles in Matlab - which we used in this case study - cannot be performed in a true parallel manner, we had to run each particle for $\Delta t = \Delta_{sample}$ time units in a single thread of execution. The condition to be able to run the PF in real time is that the computation time for running N particles over an interval of length Δ_{sample} is considerably smaller than the update time Δ_{sample} between successive measurement updates.

There is a practical problem that can appear after a few time steps of the PF algorithm. The set of particles with significant weight may no longer be sufficiently diverse to properly approximate an empirical histogram. Only a few particles have significant weights, others have become so unlikely that their weight approaches 0, which means that these particles do no longer carry useful information. Using a *resampling* procedure we can reintroduce diversity in the particles, by eliminating the particles with small weights, and by replicating particles with larger weights. There are many resampling techniques described in the literature [8], [9] that try to find a good compromise between keeping all the useful information (an unlikely particle may become likely after future observations) and avoiding time consuming calculations with very unlikely particles. The simple resampling method used in this paper is described on lines 17-22 of algorithm 1: it duplicates the $N/2$ most likely particles, and throws away the $N/2$ least likely particles.

IV. Validation of platoon based filter

The real traffic measurements used in our experiments allow thus to estimate model parameters like arrival rate and size distribution of platoons at the sources, turning ratios, and approximate values for time delays along links, and switching times of traffic lights. However no measurements on queue size evolution were available. Therefore we validated the particle filter by comparing the estimates obtained from the PF with synthetic data. Below we describe 2 such validation experiments. Some experiments were carried out by using as ground truth the platoon based model of Section II.B to generate a state trajectory, while the sensor model was used to generate the observations used as input for the PF. Robustness against modelling errors was tested by simulating the effect of an accident that according to the ground truth has a very low probability. To validate the platoon based model and to prove that it represents urban traffic with sufficient accuracy to allow particle filtering we also used as ground truth synthetic data produced by a third party microsimulator called SUMO [16].

While showing histograms would be too difficult to interpret easily, the time averages over all particles smooth out some interesting details. Therefore we represent the results by showing in each case and for all sampling times, how the queue size of the most likely particle, that is the particle with the highest weight, compares to the ground truth queue size. In order to allow easy interpretation of the figures the blocks at the bottom indicate when the traffic light is red for the direction in which the queue size is estimated.

- The first example presents how the PF is able to cope with an accident that has as effect a sudden drop of the speed, for the platoons that enter on the link: at $t = 360[sec]$ an accident blocks one lane, reducing the maximal speed along the link from $60[km/h]$ to $25[km/h]$. In order to deal with this kind of "accident" we admit that, after each Δt, the PF has a random number of particles that will embed a random drop in speed and the associated weights will be changed accordingly by assuming that an accident can take place with a probability of 10^{-5} in reality and with 10^{-2} in particles. This implements an importance sampling algorithm for dealing with the rare event of an accident. The traffic light is set to switch at each $30[sec]$. Figure 4 shows in red lines the evolution of the real queue size,

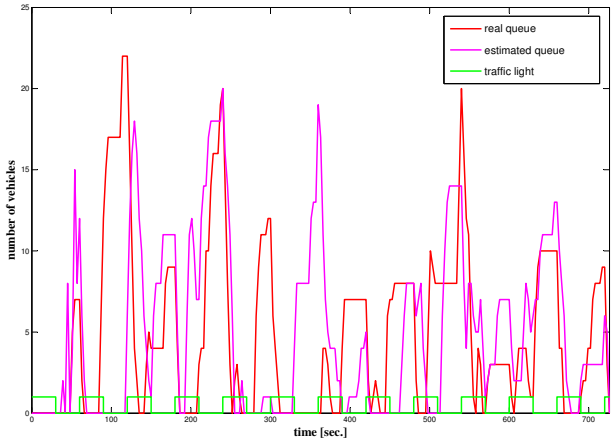

Fig. 4. Real queues (red line) vs. estimated queues (blue line) with importance sampling

and in blue lines the evolution of the estimated queue size. It can be easily observed that the PF is converging relatively fast starting around $48[sec]$ and it is able to estimate the real queue until around moment $360[sec]$ when the accident takes place. While the PF is under-estimating the real queue just after the accident the estimation is recovering after some time. The computation time was $2.2[min]$ using 202 particles, in order to analyze $12.10[min]$ of real time. The particle filter thus works about 6 times faster than real time, even with a very naive, non-optimized computer implementation.

- In the second example we have used synthetic measurements based on the induction loops sensors simulated in SUMO, for a cross-shaped network with 5 intersections interconnected by 4 links (with sensors located as in Figure 2). Model parameters (e.g. vehicle length $5[m]$, $v_{max} = 50[km/h]$ where taken from the thesis of S. Krauss [17]. The routes and the traffic were generated by using the random tool offered by SUMO. The model parameters (maximum speed, turning ratios etc.) used in the platoon based model implemented in the PF were changed accordingly (which does not invalidate our robustness claim, since these parameters can in practice be estimated on-line). The prediction errors of the queue size presented in Figure 5, using 300 particles in the PF, are still acceptable. The convergence is slower than in the first case, generally giving an overestimation of the real queue. One of the reasons can be that there is not enough noise in the

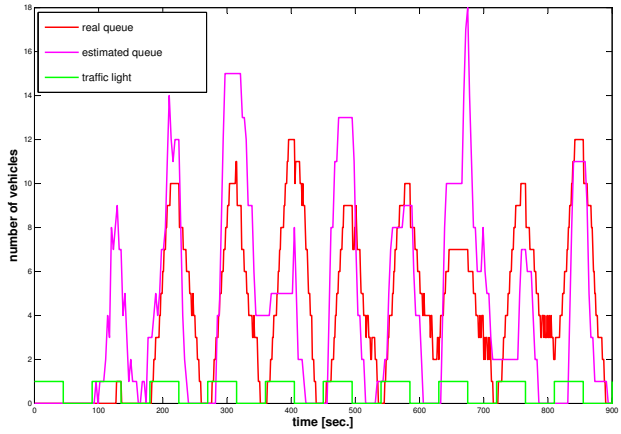

Fig. 5. Real queues generated with SUMO (red line) vs. estimated queues (blue line)

SUMO model. A computation time was 2.7[*min*], using 300 particles, was needed for a real time of 20.05[*min*].

V. Conclusions and Future Work

We introduced a discrete event system model for urban traffic based on platoons of vehicles. Validation and parameter tuning of this platoon based model uses a large set of data (collected along the N47 in Belgium). This platoon based model has been applied in a PF for estimating queue sizes and platoon location. The PF copes well with different uncertainties (sensor noise as well as model uncertainty) and allows fusion of information from different sources.

We have shown that the PF is fast enough for small networks but it is not suitable for large networks. This problem can be solved by using a distributed implementation similar to what has been presented in [10] and [11] for freeway traffic. Since the measurement update as explained in Section III only depends on the relationship between the substates in the component upstream and downstream of a sensor, a distributed implementation will only require exchanging infomration of traffic flow at boundaries. Future work will also investigate MPC controllers where the switching times of the traffic lights are selected so as to guarantee acceptable performance for all the "likely" particles (e.g. for all those particles whose weight is above an appropriately selected threshold).

Acknowledgements

This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office and EUFP7 project CON4COORD. The scientific responsibility rests with its authors.

References

[1] R. Boel and L. Mihaylova (2006) *A compositional stochastic model for real-time freeway traffic simulation*, Transportation Research B, vol.40, no.4, 319-334.

[2] L. Mihaylova and R.Boel (2004) *A particle filter for freeway traffic estimation*, Proc.43rd IEEE Conf. on Dec. & Contr., vol.40, no.4, 319-334.

[3] L. Mihaylova, R.Boel and A. Hegyi (2006) *Freeway traffic estimation within particle filtering framework*, Automatica, vol.43, no.2, 290-300.

[4] M.van den Berg,A.Hegyi,B.De Schutter,and J.Hellendoorn (2003) *A macroscopic traffic flow model for integrated control of freeway and urban traffic networks*, Proc.42nd IEEE Conf. Dec. and Control,Maui,Hawaii, pp.2774 - 2779.

[5] K. Wen, S. Qu, Y. Zhang (2009) *A Control-oriented Macroscopic Traffic Flow Model for Urban Diverse Intersections*, 2009 International Asia Conference on Informatics in Control, Automation and Robotics,pp. 62-66

[6] Y. Wang and M. Papageorgiou (2005) *Real-time freeway traffic state estimation based on extended Kalman filter: a general approach.*, Transp. Res. B, 39(2):141-167.

[7] M. Papageorgiou and J.-M. Blosseville (1989) *Macroscopic modelling of traffic flow on the boulevard Périphérique in Paris.*, Transp. Res. B, 23(1):29-47.

[8] M. Bolić, P. M. Djurić, and S. Hong (2004) *Resampling Algorithms for Particle Filters: A Computational Complexity Perspective*, EURASIP Journal on Applied Signal Processing, no. 15, pp. 2267-2277. doi:10.1155/S1110865704405149

[9] R. Douc and O. Cappe (2005) *Comparison of resampling schemes for particle filtering*, Image and Signal Processing and Analysis, 2005. ISPA 2005. Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis, pp. 64 - 69.

[10] L. Mihaylova, A. Gning, V. Doychinov and R. Boel (2009) *Parallelised Gaussian Mixture Filtering for Vehicular Traffic Flow Estimation*, Proceedings INFORMATIK2009, September 2009, Luebeck, Germany

[11] A. Hegyi, L. Mihaylova, R. Boel, Z. Lendek (2007) *Parallelised Particle Filtering for Freeway Traffic State Tracking*, Tutorial Session "Motorway Traffic Surveillance and Control II" (TuD15), European Control Conference ECC07, July 2007, Kos, Greece

[12] Y. Wang, M. Papageorgiou, and A. Messmer (2003) *Motorway traffic state estimation based on extended Kalman filter*, Proceedings of the European Control Conf., UK, 2003

[13] A. Doucet, N. Freitas, and Eds. N. Gordon (2001) *Sequential Monte Carlo Methods in Practice.*, New York: Springer-Verlag, 2001.

[14] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P-J. Nordlund (2002) *Particle filters for positioning, navigation and tracking.*, IEEE Transactions on Signal Processing, 50(2):425-437, 2002.

[15] X. Sun, L. Muñoz, and R. Horowitz. (2003) *Highway traffic state estimation using improved mixture Kalman filters for effective ramp metering control.*, In Proc. of 42th IEEE Conf. on Decision and Control, pages 6333-6338. , USA, 2003.

[16] D. Krajzewicz, G. Hertkorn, C. Rössel, P. Wagner. 2002-2005. SUMO Homepage. `http://sumo.sourceforge.net`.

[17] S. Krauss *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics.*, Hauptabteilung Mobilität und Systemtechnik des DLR Köln. ISSN 1434-8454, 1998.

[18] N. Marinică, R. Boel. (2011) *Particle filter for platoon based model of urban traffic.*, The 4th WSEAS International Conference on Urban Planning and Transportation (UPT 11), Greece, 2011.

[19] G. Vigos, M. Papageorgiou, Y. Wang (2008) *Real-time estimation of vehicle-count within signalized links* Transportation reseach Part C, 16(1):18-35, 2008.

[20] C. Daganzo (2007) *Fundamentals of Transportation and Traffic Operations* Emerald Group Publishing, 2007.