Flow Reconstruction for Data-Driven Traffic Animation

David Wilkie*
University of North Carolina

Jason Sewall†
Intel Corporation

Ming Lin‡
University of North Carolina

Figure 1: *The divided highway is populated with vehicle traffic using our flow reconstruction method. Individual virtual cars are animated and correspond to real-world traffic conditions measured by road sensors. The vehicle motions are simulated on a GIS-derived domain and overlaid on a public domain aerial image from the U.S. Geological Survey.*

Abstract

'Virtualized traffic' reconstructs and displays continuous traffic flows from discrete spatio-temporal traffic sensor data or procedurally generated control input to enhance a sense of immersion in a dynamic virtual environment. In this paper, we introduce a fast technique to reconstruct traffic flows from in-road sensor measurements or procedurally generated data for interactive 3D visual applications. Our algorithm estimates the full state of the traffic flow from sparse sensor measurements (or procedural input) using a statistical inference method and a continuum traffic model. This estimated state then drives an agent-based traffic simulator to produce a 3D animation of vehicle traffic that statistically matches the original traffic conditions. Unlike existing traffic simulation and animation techniques, our method produces a full 3D rendering of individual vehicles as part of continuous traffic flows given discrete spatio-temporal sensor measurements. Instead of using a color map to indicate traffic conditions, users could visualize and fly over the reconstructed traffic in real time over a large digital cityscape.

CR Categories: G.1.1 [Mathematics of Computing]: Interplation—Smoothing; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

Keywords: Virtual Traffic, Interactive Techniques, Urban Scenes

Links: DL PDF WEB

*wilkie@cs.unc.edu
†jason.sewall@intel.com
‡lin@cs.unc.edu

ACM Reference Format
Wilkie, D., Sewall, J., Lin, M. 2013. Flow Reconstruction for Data-Driven Traffic Animation. ACM Trans. Graph. 32, 4, Article 89 (July 2013), 10 pages. DOI = 10.1145/2461912.2462021 http://doi.acm.org/10.1145/2461912.2462021.

DOI: http://doi.acm.org/10.1145/2461912.2462021

1 Introduction

Numerous efforts have been devoted to acquiring and visualizing "digital urbanscapes". Over the last decade, there has been considerable progress on multiple fronts: acquisition of imagery and 3D models using improved sensing technologies, real-time rendering, and procedural modeling. For example, aerial imagery of most cities is used in Google Earth and Microsoft Virtual Earth. The problem of reconstructing 3D geometric models from videos and scanners has been an active area of research. Similarly, many efficient techniques have been proposed to stream imagery and geometric data over the Internet and display them in real time on high-end workstations or handheld devices. However, all these efforts are limited to capturing, displaying, or modeling predominantly static models of urbanscapes and do not include dynamic elements, such as traffic. The realism of a virtual urbanscape in a digital globe system can be considerably enhanced by introducing such intrinsic dynamic elements of an urban landscape.

As VR applications in flight and driving simulators [Cremer et al. 1997; Donikian et al. 1999; MIT 2011; Pausch et al. 1992; Thomas and Donikian 2000; Wang et al. 2005] for training have evolved from its earlier single-user VR system into online virtual globe systems and distributed, networked gaming, the demand to recreate large-scale traffic flows, possibly driven by traffic sensor data from the real-world observations, has emerged. The concept of 'virtualized traffic' was first introduced in [van den Berg et al. 2009; Sewall et al. 2011a] to create dynamic vehicle flows based on real-world traffic sensor data to enhance the sense of immersion for a virtual urbanscape. We propose an efficient technique for reconstructing 'virtualized traffic' from continuous streams of traffic data *either* from in-road sensor measurements *or* procedurally generated control/user input. This approach consists of (1) a data analysis stage, in which the traffic state is estimated via an ensemble Kalman smoothing process and a macroscopic model for representing aggregate traffic flow, and (2) a visualization stage, in which a detailed reconstruction of the traffic flow is rendered and displayed. The final rendering is a 3D visualization of vehicle traffic as it might appear to a camera, but reconstructed wholly from sparse, discrete traffic data. This technique could be used by someone planning a trip [Wilkie et al. 2011], enabling them to do a fly-over or see the traffic from a driver's perspective; it could be used

by city planners and traffic controllers to see an integrated visualization of traffic over a large metropolitan area; and it could be used to populate virtual worlds to create a more immersive experience and make vehicle flows more realistically reflecting the real-world traffic. Unlike video recordings from individual cameras, our method uses existing infrastructure, requires very little bandwidth for communication, and allows for large-scale integrated views in any VR application.

Traffic is an aggregate of individual vehicles, each moving at a velocity dependent on the surrounding vehicles and conditions too numerous to fully list (or perhaps know), such as the time of day, weather, signage, road work, et cetera. We consider an estimate of traffic flow over a span of roadway to be the combination of a density field and a velocity field. These fields model the macroscopic conditions of traffic flow at a particular instant.

Sensors come in several forms. The most widespread are called loop detectors, which are placed on roadways and record attributes for every vehicle that passes. Another form of sensor is within the vehicle itself: cell phones and GPS devices can monitor the speed of the vehicle and report it along with their position. Finally, video cameras can also be used to monitor traffic, providing full trajectories for all observed vehicles. In this paper, we focus on in-road sensing, such as loop detectors and video cameras. These sensors can detect when a vehicle has passed and can provide such information as the speed the vehicle is traveling, the length of the vehicle, whether the vehicle is a truck or car, *et cetera*. Typically, these sensors batch their transmissions and send aggregate data for all vehicles within a time span. Even if the transmissions are instantaneous, they are spatially and temporally discrete readings. Although our method assumes the available traffic sensor data from loop detectors and video cameras in our implementation, the overall computational framework can be easily generalizable and applicable to other forms of sensor measurements.

Main Result. We present an efficient technique that enables the creation of a detailed 3D traffic reconstruction from sparse traffic data. Our method features a traffic state estimate phase, in which an ensemble Kalman smoother and a continuum traffic simulator are used to create an estimate of velocity and density fields over the entire road network. Our method uses this estimate as a control for an agent-based traffic simulator to create the 3D animation of individual vehicle motion. The agent-based system is controlled by automatically assigned boundary conditions, the estimated velocity field, and a simplified leader-follower relationship. Finally, the output is a 3D visual display of traffic flow consistent with the original traffic pattern measured by the sensors. It is important to note that our method is not a traffic simulation, but a reconstruction of vehicle flows that corresponds to measured traffic conditions as observed in the real world. Furthermore, the resulting 3D traffic animation can be interacted with and respond to user control and manipulation in a virtual environment.

The rest of the paper is organized as follows. Section 2 provides related research; state estimation methods and traffic simulation are covered, as well as other approaches to traffic reconstruction. Section 3 describes our approach, including both the state estimation phase and data reconstruction phase. Section 4 provides statistical and visual results from our approach using real traffic data sets. Section 5 concludes the paper with discussion and future work.

2 Previous Work

Research on traffic estimation has a long history, dating back to the 1970s (see an early review by Cremer [1991]). Recent projects have similar goals and approaches: they want a high-level estimation of macroscopic traffic quantities, and they use a filtering algorithm to estimate the state based on incoming sensor data and a model of traffic dynamics. Hegiy et al. [2007] discusses parallelization of a Particle Filter approach to traffic state estimation. Jacquet et al. [2005] present an approach to handle the challenge of shock waves in traffic state estimation. Jacquet et al. [2006] describe an approach that allows gradient descent to be used on traffic simulation models, with applications to traffic state estimation. Sau et al. [2007] discuss real-time state and parameter estimation using particle filters. Wang and Papageorgiou [2005] present a detailed investigation of real-time traffic state estimation using the extended Kalman filter. Finally, Work et al. [2010] develop a velocity based traffic model in order to estimate traffic using cell phone signals.

The majority of these traffic estimation approaches rely on a Kalman filtering approach. One particular class of filters of note is the *Ensemble Kalman Filter* (EnKF). This is the filter of choice for estimation of systems involving (very) high dimensional states. It has been used for weather prediction [Houtekamer and Mitchell 2001], ocean current estimation, as well as traffic estimation [Work et al. 2008; Work et al. 2010]. Similar to a particle filter, the EnKF propagates a set of samples of states that model a probability distribution. For the EnKF these samples form the *ensemble* for the system. Unlike a particle filter, the EnKF is correct in the limit (infinite number of samples) only for linear systems with Gaussian noise. However, the fact that its running time is only *linear* in the dimension of the state, rather than *cubic* as for other filters, makes its use advantageous for non-linear systems as well, if it can be reasonably assumed that the distribution of the state has a single mode. While these works provide an estimation of the entire traffic flow on a single highway, they offer little detail to visualize the actual vehicle traffic condition.

An important component in state estimation is a model of traffic dynamics. A common model is by Lighthill and Whitham [1955] and Richards [1956], called the LWR model, which was a seminal macroscopic traffic dynamics model. Traffic simulation is well studied, and the amount of literature is too vast to summarize here. A detailed survey can be found in [Helbing 2001]. Some notable methods include the ARZ model, by Aw and Rascle [2000] and Zhang [2002], which is what this work and earlier continuum simulation [Sewall et al. 2010] are based on, and the model of Papageorgiou *et al.* [1990], which was used to model the traffic dynamics in the traffic state estimation work in [Wang and Papageorgiou 2005].

This work builds on recent advances in traffic simulation. Wilkie et al. [2012] propose a method for extrapolating GIS road networks into a C^1 smooth road network representation suitable for traffic simulation and animation. We use this method to construct the entire road network for running our traffic simulation on. We also adapt the visualization technique by Sewall et al. [2010] on continuum traffic simulation for a large-scale vehicle traffic visualization using animated, individual vehicle representations instead of particle tracers or vector fields commonly performed in flow visualization [Laramee et al. 2004]. However, [Sewall et al. 2010] is only for forward simulation, not for data-driven animation or traffic reconstruction. Our 3D traffic visualization take into account of vehicle kinematics and dynamics for lane changing and visualization of individual cars. Our method differs from [van den Berg et al. 2009] and [Sewall et al. 2011a], which reconstruct plausible trajectories for individual cars based on microscopic boundary conditions using priority-based motion planning techniques. Such techniques can quickly become intractable as the resolution of discretization in search space increases [Sewall et al. 2011a]. In contrast, we use an agent-based simulation with controlled temporal and spatial constraints that ensure the cross flows among multiple lanes are consistent with the high-level state estimation of the overall traffic flow. We also assume realistic sensor models in our implementation.

Most recently [Sewall et al. 2011b] proposed a hybrid simulation method that uses both macroscopic and microscopic simulators simultaneously operating on distinct regions of a road network. Although both continuum and discrete simulation methods are used in this work, our method differs by first applying the macroscopic method for overall flow estimation and then the particle system simulation to reconstruct the detailed vehicle flow; it then refines the global state estimation that accounts for the difference between the estimated and simulated states of the traffic system due to cross flows among multiple lanes. Our selective, sequential use of each type of simulation methods best exploits the individual method's strength in our iterative estimation-reconstruction-refinement framework and does not apply both methods at the same time to different locations as in [Sewall et al. 2011b].

One simple and straight-forward method to visualize the current traffic condition is direct playback of the captured video data from the road networks. But, the collective loads on the transmission bandwidth and the number of camera installations would make such an obvious approach impractical; furthermore, it cannot offer other computational benefits for driver assistance, navigational aid, and important decision-making analysis critical to other applications. To the best of our knowledge, this work is the first method that offers real-time visualization of vehicle traffic reconstructed directly from temporal-spatial data readily available from existing in-road sensors on the road networks. It does not require additional identification of individual vehicles, such as [Sewall et al. 2011a; van den Berg et al. 2009], to visualize the current vehicle traffic condition based on sensor measurements.

3 Approach

Our approach estimates traffic conditions based on a stream of traffic data. The estimate is created by a smoothing process involving an Ensemble Kalman Smoother and a macroscopic traffic simulator. We use the estimate to drive an agent-based simulation to produce a visualization of the traffic conditions. An overview of the approach can be seen in Fig 2.

3.1 Preliminaries

Sensors: Our model assumes that the traffic conditions are monitored by sensors. We assume these sensors are discrete spatially and provide measurements at discrete time intervals. These measurements are assumed to be an estimate of the density and velocity, or flow, of the traffic over the time interval. These sensors can come in multiple forms as long as certain assumptions are satisfied. Specifically, we assume that we know the location of the sensor, and that the position of the sensor is fixed. We assume that the data the sensors provide can be transformed into a density and velocity estimate. A prominent form of sensor of this type is the loop detector. This form of sensor provides volume and velocity information, from which the density of traffic can be calculated. Another form of sensor that could satisfy these requirements are cameras, from which the density and velocity of traffic at a point on the roadway could be estimated.

Road Networks: The domain of a traffic simulation is a road network. These networks can vary in complexity from being a single lane to being a vast network of roads with multiple lanes, intersections, highway ramps, overpasses, sources, sinks, and other features. Our method assumes a multi-lane highway as its domain. We follow [Wilkie et al. 2012] in creating the road representation from available GIS data. The created geometry is C^1 smooth and ensures that car movements will satisfy the *simple car* kinematic constraints [van den Berg et al. 2009; Sewall et al. 2011a]. Each lane has relations defined to allow merging both between neighboring lanes as well as between the highway and ramps. The geometry of each lane is discretized for the purposes of macroscopic simulation as well as state representation. The lanes have associated lengths, and cars moving along the lanes have positions in a local 1D coordinate frame, where the beginning of the lane is at position 0 and the end is at position *length*. The cell containing a particular position p will be referred to as *cell(p)*.

3.2 State Estimation

Our approach assumes traffic data are spatially and temporally sparse and therefore describe only a small fraction of the overall traffic conditions. Under such assumptions, it is necessary to estimate the full traffic state given the sensor measurements. We achieve this estimation via a state estimation process, which makes use of all available sensory information, as well as a macroscopic model of traffic dynamics. An example of the ground truth and resulting estimate can be seen in the Appendix.

The full traffic conditions, as we described previously in Section 1, involve the states of all the individual vehicles. To abstract these conditions to the form of a *state* of traffic, we must make an assumption about the dynamics of traffic, as different dynamics models will require different state formulations in order to fully specify how the traffic will evolve. We assume a macroscopic model described in [Sewall et al. 2010], based on the equations of Aw and Rascle [2000] and Zhang [2002]. Following Lebacque [2007], we refer to this as the Aw-Rascle-Zhang (ARZ) model. This traffic model describes the evolution of aggregate traffic statistics, density and velocity, along lanes and makes use of a parameter for determining the speedlimit, $v_{\max}$, and a parameter γ to define a relationship between the velocity and the density. Following this, the state of traffic will be a vector,

$$\mathbf{x} = (\rho_1, y_1, \rho_2, y_2, ..., \rho_N, y_N, \gamma, v_{\max}) \tag{1}$$

where ρ_i is the density and y_i is the relative velocity, defined below, of the i^{th} *cell* of the lane; a single lane with n discrete cells will have $2n+2$ degrees of freedom.

3.2.1 Flow Reconstruction

To create a state estimate, we do a *smoothed reconstruction* of the traffic using sensor measurements and a macroscopic traffic simulator. This reconstruction uses the Ensemble Kalman Smoothing (EnKS) method [Evensen 2003]. The estimation process maintains an *ensemble* of states for each timestep, and each ensemble member is updated sequentially when a new sensor measurement is received. The update involves advancing the ensemble members via simulation and applying corrections based on the difference between the received measurements and the current state. The estimated state of the traffic at time $t_i, i \in [0,N)$ is the mean of the ensemble of potential states for time t_i after the sensor measurement at t_{N-1} has been received.

The EnKS has two principle components, a *motion model* and an *observation model*. The motion model,

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \mathbf{m}_t), \quad \mathbf{m}_t \sim \mathcal{N}(0, I), \tag{2}$$

describes the evolution of the dynamic system. It propagates the state $\mathbf{x}_{t-1}$ to the state $\mathbf{x}_t$ given noise $\mathbf{m}_t$. This particular formulation assumes that the state depends only on the previous state and noise, not control inputs. The observation model,

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{n}_t), \quad \mathbf{n}_t \sim \mathcal{N}(0, I), \tag{3}$$

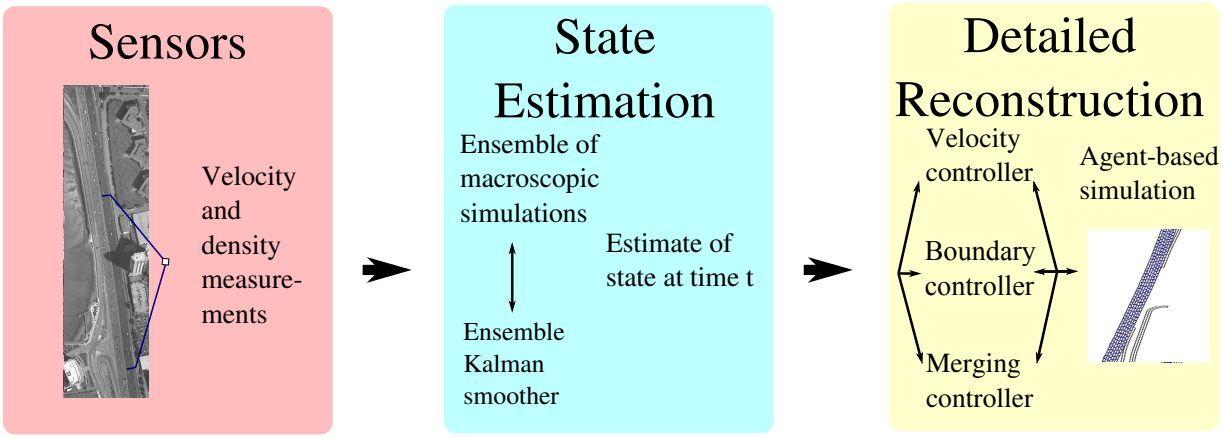

Figure 2: *A schematic view of the approach.*

relates the simulated dynamic states, $\mathbf{x}_t$ to real world measurements, given measurement noise $\mathbf{n}_t$.

These models are used to propagate an ensemble of states, $\mathscr{X}_t = \{\mathbf{x}_t^0, \ldots, \mathbf{x}_t^{M-1}\}$ that represents the distribution of the unknown true state of the system. At each time interval, both models are calculated, and the result is used to update the ensemble. This is shown in 1, where the $\mathbf{m}_t^i$ is drawn randomly and independently from $\mathscr{N}(0,I)$, $\mathbf{n}_t^i$ is drawn randomly and independently from $\mathscr{N}(0,I)$, and $\mathbf{z}_t$ is the obtained measurement. Note that $\hat{\mathbf{z}}_t = E(\mathbf{z}_t)$, $\hat{\mathbf{x}}_t = E(\mathbf{x}_t)$, $\Sigma_t = Var(\mathbf{z}_t)$, and $\Gamma_t = Cov(\mathbf{x}_t, \mathbf{z}_t)$.

Algorithm 1: EnKS for Traffic State Estimation

Input: Traffic sensor measurements $z_1 \ldots z_{t_n}$, ARZ simulator in f, observation model in h, initialized ensemble $\mathbf{x}_0^0 \ldots \mathbf{x}_0^{m-1}$, noise vectors $\mathbf{m}_t^0 \ldots \mathbf{m}_t^{m-1}$ and $\mathbf{n}_t^0 \ldots \mathbf{n}_t^{m-1}, \forall t \in 1 \ldots t_n$

Output: Traffic state estimates $\hat{\mathbf{x}}_t, \forall t \in 1 \ldots t_n$

for $t \in 1 \ldots t_n$ **do**
  for $i \in 0 \ldots M-1$ **do**
    `// Motion model`
    $\mathbf{x}_t^i \leftarrow f(\mathbf{x}_{t-1}^i, \mathbf{m}_t^i)$;
    `// Observation model`
    $\mathbf{z}_t^i \leftarrow h(\mathbf{x}_t^i, \mathbf{n}_t^i)$;
  `// Analysis`
  $\hat{\mathbf{z}}_t \leftarrow \frac{1}{M} \sum_{i=0}^{M-1} \mathbf{z}_t^i$;
  $\Sigma_t \leftarrow \frac{1}{M-1} \sum_{i=0}^{M-1} (\mathbf{z}_t^i - \hat{\mathbf{z}}_t)(\mathbf{z}_t^i - \hat{\mathbf{z}}_t)^T$;
  for $j \in 1 \ldots t$ **do**
    $\hat{\mathbf{x}}_j \leftarrow \frac{1}{M} \sum_{i=0}^{M-1} \mathbf{x}_j^i$;
    $\Gamma_j \leftarrow \frac{1}{M-1} \sum_{i=0}^{M-1} (\mathbf{x}_j^i - \hat{\mathbf{x}}_j)(\mathbf{z}_t^i - \hat{\mathbf{z}}_t)^T$;
    $K_j \leftarrow \Gamma_j \Sigma_t^{-1}$;
    for $i \in 0 \ldots M-1$ **do**
      $\mathbf{x}_j^i \leftarrow \mathbf{x}_j^i + K_j(\mathbf{z}_t - \mathbf{z}_t^i)$,;
`// Final output`
for $t \in 1 \ldots t_n$ **do**
  $\hat{\mathbf{x}}_t \leftarrow \frac{1}{M} \sum_{i=0}^{M-1} \mathbf{x}_t^i$;

In the sequel, we discuss the details of this process, including ensemble initialization, the motion and observation model formulations, and issues concerning noise and tuning.

3.2.2 Ensemble Initialization

We initialize each ensemble member, $\mathbf{x}_0^i$, with random values for density, ρ, relative velocity, y, speedlimit, $v_{\max}$, and the fundamental diagram parameter, γ. $v_{\max}$ is drawn uniformly from $[10, 60]$ meters per second; γ is drawn uniformly from $[0.05, 0.9]$. The density for the first cell is drawn uniformly from $[0, 0.9]$, as we consider the maximum density, 1, to be unachievable in real world scenarios. The velocity, v, for the first cell is drawn uniformly from $[0, v_{\max}]$, from which a relative velocity is calculated.

The density and velocity for subsequent cells is the result of a dynamic random walk with bounded $\dot{\rho}$, $\ddot{\rho}$, $\dot{v}$, and $\ddot{v}$. The created density and velocity fields are C^2 when they are within the imposed bounds on ρ and v. While we discuss the creation of the state in terms of the velocity, it is the subsequent relative velocity that is stored in the state.

3.2.3 Motion Model

As discussed in the previous section, the motion model propagates each ensemble member forward to time t given the state at time $t-1$. The state, as described above, is defined to fully specify the evolution of traffic using the ARZ simulation formulation. For each iteration, each ensemble state x_{t-1}^i is evolved forward using the motion model. Each lane is simulated independently, with the density and velocity values for the 0^{th} and N^{th} cells being held constant to implement the boundary conditions. The duration of the simulation is defined by Δt_{filter}, i.e. the frequency with which traffic data are received. After each simulation run, Gaussian noise is added to each member of the state, as described below.

ARZ system of equations The macroscopic simulation component of our method is based on the nonlinear hyperbolic system of partial differential equations proposed by Aw and Rascle [2000] and Zhang [2002]; this system describes the motion of vehicles along a lane in terms of vehicle density ρ, velocity v, and the derived quantity y, the so-called *relative velocity*. The system is:

$$\mathbf{q}_t + \mathbf{f}(\mathbf{q})_x = 0, \ \mathbf{q} = \begin{bmatrix} \rho \\ y \end{bmatrix}, \ \mathbf{f}(\mathbf{q}) = \begin{bmatrix} \rho v \\ y v \end{bmatrix} \tag{4}$$

In the parlance of hyperbolic PDEs, Eq. (4) is in *conservation form*; the column vector $\mathbf{q} = [\rho, y]^\mathsf{T}$ describes quantities whose totals in the whole system change only due to what flows in and out of boundaries. Conservation forms are convenient for discovering im-

portant properties of and finding solutions to hyperbolic systems, and this motivates the introduction of the relative velocity y.

Relative velocity It makes intuitive sense that the density of vehicles ρ must be conserved (its sum over the system reflects the total quantity of vehicles therein), but we should not expect that the velocity v should directly be conserved — were this so, whenever a vehicle slowed down, another would have to accelerate!

Thus, the concept of relative velocity y is introduced, which roughly describes how fast vehicles are traveling compared to the 'optimal velocity' for a given density. Specifically, y is related to ρ and v as follows:

$$y(\rho, v) = \rho\left(v - v_{\text{eq}}(\rho)\right) \tag{5}$$

The quantity $v_{\text{eq}}(\rho)$ is the *equilibrium velocity* for ρ — the largest velocity that may be achieved at a given ρ. This is closely related to an important concept in traffic flow, the *fundamental diagram*, which is a curve that plots the *flux* of traffic (i.e. the throughput of vehicles ρv) as a function of increasing density [Zhang 2002]. Fundamental diagrams differ from flow to flow, but invariably theoretical models and empirical observations show that they are convex functions of ρ with a single, unique maximum; this fits intuition — where $\rho = 0$, there is clearly no flux, and as ρ approaches 1 (bumper-to-bumper traffic), velocity approaches 0 and flux falls back to zero.

In the ARZ model, $v_{\text{eq}}(\rho)$ is defined as follows:

$$v_{\text{eq}}(\rho) = v_{\max}(1 - \rho^{\gamma}) \tag{6}$$

where the $v_{\max}$ is the speedlimit of the lane and $\gamma \in (0,1)$ is a parameter that characterizes the fundamental diagram.

Numerical macroscopic solutions To obtain numerical solutions to Eq. (4), we use the Finite Volume Method with Riemann solver specific to the equations.

A lanes is partitioned into N discrete cells of length Δx. The discrete quantity $\mathbf{Q}_i = [\rho_i, v_i]^{\mathsf{T}}$, $i \in \mathbb{Z}[0,N)$ represents the average value of $\mathbf{q}$ over cell i.

To advance the solution from time t_n to t_{n+1}, we use forward Euler integration:

$$\mathbf{Q}_i^{n+1} = \mathbf{Q}_i^n - \frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i-}^n + \mathbf{F}_{i+}^n\right) \tag{7}$$

where $\mathbf{Q}_i^n$ is the average value of $\mathbf{q}$ in cell i at time n, $\Delta t = t_{n+1} - t_n$ and the fluxes $\mathbf{F}_{i-}^n$ and $\mathbf{F}_{i+}^n$ are the average value of $\mathbf{f}(\mathbf{q})$ from t_n to t_{n+1} at the interface between i and the neighboring cells $i-1$ and $i+1$, respectively (i.e. $\mathbf{F}_{i^-}^n = \mathbf{F}_{i-1^+}^n$ and $\mathbf{F}_{i^+}^n = \mathbf{F}_{i+1^-}^n$).

In practice, for nontrivial $\mathbf{f}$ such as that found in Eq. (4), exact computation of $\mathbf{F}_{i-}^n$ and $\mathbf{F}_{i+}^n$ is not possible and we must settle for an approximation; the most robust such approximations are found by solving the so-called *Riemann problem* at each interface between adjacent cells to determine how the solution is evolving there. The computation of these Riemann problems is the most computationally expensive portion of the macroscopic solution procedure, but is critical to obtain correct, stable solutions in the presence of nonlinear phenomena such as shocks, rarefaction, and "vacuum" states.

A detailed description of the Riemann problem for the ARZ system of equations and the appropriate accompanying Riemann solver can be found in Appendix B of [Sewall 2011].

3.2.4 Observation Model

The observation model creates a measurement z_t^i of the simulation state x_t^i in order to compare the simulation with the sensor measurements. We create a vector containing density and velocity measurements of the simulation. For each real-world sensor location p, we return the ρ, v pair for $cell(p)$. These measurements are perturbed with Gaussian noise, as described below.

3.2.5 Tuning and Noise

After each ensemble member has been propagated by the motion model, zero-mean Gaussian noise is added to each cell. For each cell, the noise is scaled to account for the differences between different parts of the states. Specifically, the γ and v_{max} elements of the state receive more noise than the density and relative velocity elements, and the boundary elements receive more noise than the interior elements. Additionally, as we have topological information about the simulation domain, we can enhance the model by transforming the noise at points according to the road network. For example, we allow a wider range of noise at on ramps and off ramps (respectively additive noise and subtractive noise) to allow the system to account for cars entering or leaving the lane. Values for these noise weights can be found in 4.4.

3.3 Detailed Reconstruction

In the state estimation phase, we generate estimates for the average state of traffic over a time interval, defined as the time between two subsequent sensor measurements. These state estimates are created with a macroscopic simulation, implying no individual cars exist to be visualized. Additionally, the state estimation is done independently for each lane of the road network.

In this section, we present our approach to creating an agent-based simulation that matches the given macroscopic state estimate. This stage simulates traffic on all the lanes of the roadway simultaneously and uses controlled merging between them. The end result of this process is an animation that provides real-time visualization of the traffic flow, where the output is a collection of states for individual cars. The simulation is controlled via boundary conditions, parameter fitting, and merging.

3.3.1 Vehicle Instantiation

We must provide an initial population of vehicles along each lane. To do this, we use a straight-forward approach of creating cars as evenly spaced as possible to satisfy the density specified in the state estimate. The result of this step may appear too uniform to be mistaken for real traffic, but our primary concern here is to show that the statistical properties of the original traffic flow are captured. As such, we wish to introduce as little additional noise as possible.

To achieve this for lane j, we take its state estimate at the first considered timestep, $\mathbf{x}_0$. Let ρ_0 be the vector of densities in $\mathbf{x}_0$, and $\mathbf{v}_0$ be the velocities derived from the relative velocities, $v_{\max}$, and γ of $\mathbf{x}_0$. Let p be a position in the local 1D coordinate frame of the lane, as described in Section 3.1. We initialize p to 0, the beginning of the lane. Then, iteratively, we calculate the *separation distance* that is implied by the $cell(p)^{th}$ member of ρ_0, i.e. the density estimate for that part of the lane. The separation distance is defined as $\frac{1}{\rho_p} car\ length - car\ length$. The separation distance is added to p, and a car is added to the roadway at p. The cars initial velocity is set to the $cell(p)^{th}$ member of v_0.

3.3.2 Particle Advection

We use an agent-based simulation to visualize the individual cars that make up the traffic flow. Each car is advected using the velocity field estimated previously. The velocity field estimates are lane-specific and constant during a filter interval. The field is discretized into cells, each corresponding to a cell of the macroscopic simulation (as discussed in 3.1). To increase the heterogeneity of the simulation, each car can optionally be given a velocity scaling factor, drawn from a bounded normal distribution, to bias its preferred speed.

To allow for accurate integration over large timesteps, we calculate a velocity for each car that is the weighted combination of the cell velocities through which the car will pass during the timestep. Each velocity is weighted by the amount of time the car will spend within the corresponding cell.

As the velocity field is constant over a time interval, it is possible for cars to come into collision. To prevent this, we can restrict the velocity applied to a car in *cell(p)* as $v = \min(v_p, \Delta x - s_1 + v_l)$, where v is the velocity applied to the car, v_p is the velocity of the estimated velocity field, Δx is the distance between the two cars, s_1 is a minimum separation distance (including the car length), and v_l is the velocity of the leader car. Thus a car cannot take a velocity that would result in it being a distance less than $s1$ from the leading car.

3.3.3 Boundary Conditions

As the agent-based simulation advances, cars will leave the network and new cars need to be added to visualize the traffic flow. The inflow requirement is defined a flow rate for each lane as $r_l = \rho_l^s v_l^s$, where ρ_l^s and v_l^s are the density and velocity of the first cell in the state estimate for the lane. We create cars uniformly to satisfy this rate as follows. From the density component, ρ_l^s, we can derive the separation distance s_l, as described in 3.3.1. During each timestep, the last car in a lane will be at position p. We create cars such that the i^{th} created car is at position $p - s_l * i$. The cars are created until there is no longer any space in the lane. The velocity component of the flow rate is satisfied implicitly: the time required for sufficient free space to be created is dependent on the velocity that the created cars are moving.

Creating cars in this manner can match the flow rate as well as allow large time steps if desired, however the traffic created can appear too uniform to be realistic. To prevent this, we can add bounded zero-mean Gaussian noise, n_s, to the separation distances used for car creation. The noise is bounded such that $n_s > s_m + s_{mT} v_l^s$, where s_m is a minimum distance including the car length and some separation distance, s_{mT} is a minimal stopping time that determines, with v_l^s, a dynamic minimal distance. This constraint is derived from the Intelligent Driver Model [Treiber et al. 2000], which is a microscopic traffic model based on a leader-follower equation.

3.4 Merging

Modeling traffic merging is an area of ongoing research that involves driver decision making as well as a complicated interplay of dynamics, kinematics, and multi-agent reactions. For visually realistic and detailed animations, a merge must be considered as having a time duration larger than a simulation timestep. This implies cars must be capable of being in the state of *merging*, and other cars need to react appropriately to merging cars. Further, for controlled traffic animation, we need to constrain and direct merging to bring the animation in line with the target state, while allowing users to specify a level of merging activity to achieve their desired animation goal.

Our approach to controlled merging takes into account the accumulated error in each lane's density field. Starting from the most downstream point of the highway, density error is aggregated for each lane. If the absolute error is greater than one car, the corresponding cell of the road is marked as a merge point. Cars at these merge points can switch between lanes to account for the density errors, as long as the agent-based kinematic and dynamic conditions are satisfied.

We use additional decision criteria to determine if a car should merge based on the local dynamics of the system. A car will not merge if there is not enough space for it, if it would cause too great a deceleration in its target lane, if, by merging, it would decelerate itself, or if there is already a car merging into the lane in the neighborhood upstream. These rules ensure that merging does not detract from the natural flow of traffic or create unrealistic looking situations.

It is important to note that while lane changing and merging are departures from the ARZ system of equations, the dynamics that result can be formally accounted for in the underlying equations as source terms on the right-hand side; indeed, one such source term for relaxation of relative velocity has traditionally used to promote vehicle acceleration in the presence of headway. In particular, the boundary integral of the multi-lane road still obeys conservation in the first family; vehicles (density) is not lost or gained.

We implement these criteria using rules defined in terms of leader, follower, target lane, and the dynamic and geometric characteristics of the cars.

Definition 1. target lane. For a car a, an adjacent lane is one reachable by a right or left merge. A target lane is an adjacent lane that is a is considering merging into.
Definition 2. leader. For a car a and a lane l that either contains a or is adjacent to a lane that contains a, we define the leading cars as $LC = \{c | c.p > a.p_L\}$, and the leader to be $b \in LC$, s.t. $b.p <= c.p$ forall $c \in LC$.
Definition 3. follower. For a car a and a lane l that either contains a or is adjacent to a lane that contains a, we define the following cars as $FC = \{c | c.p < a.p_L\}$, and the follower to be $b \in FC$, s.t. $b.p >= c.p$ forall $c \in LC$.
Definition 4. f_{LF}**.** A leader-follower function that returns an acceleration value given the positions and velocities of a leader car and a follower car.

The four criteria based on these terms are defined as follows.

Criterion 1. For a car a, let b and c be the leader and follower in the target lane respectively. A merge is forbidden if $((b.p - a.p) < d_{carspace}) or ((a.p - c.p) < d_{carspace}) or (b.p - c.p < d_{clear})$, where $d_{carspace}$ is the length of a car and a small buffer and d_{clear} is the *gap clearance* required for a safe merge.
Criterion 2. For a car a, let b be the leader in the target lane. A merge is forbidden in $f_{LF}(a.p, b.p, a.v, b.v) < 0$.
Criterion 3. For a car a, let b be the follower in the target lane. A merge is forbidden if $f_{LF}(a.p, b.p, a.v, b.v) < a_{disrupt}$, where $a_{disrupt}$ is an acceleration limit.
Criterion 4. For a car a, let b be the follower in the target lane. A merge is forbidden if b is in the state of *merging* to the target lane.

Once a car a decides to merge, the actual merge operation must take place over a simulation time interval. This involves moving the car a from its current lane to its target lane in a manner that respects both kinematic and dynamic constraints. In regard to the kinematic constraints, merges are often geometrically modeled as a trajectory resulting from a constant rate of change in the steering angle, i.e. turning the steering wheel at a constant rate into the turn and then at a constant rate out of the turn. The resulting path is a Clothoid (or

Euler or Cornu) curve, for which no analytical expression exists. As a substitute, we use a polynomial approximation of the curve. In regard to the dynamic context, once a begins a merge, it can be said to belong to both its current lane and its target lane. However, a conservative approach to reconciling these two control influences leads to unrealistic traffic flow and congestion. We therefore allow a merging car to safely ignore its leader once it has passed the midway point of its merging curve.

4 Results

We have implemented our techniques presented here and tested them on traffic data from publicly available online sources.

To demonstrate our method, we reconstruct virtualized traffic on a segment of highway. The sensor data used for these experiments is from the Next Generation Simulation (NGSIM) program [NGS 2013]. The roadway on which the data are measured is a stretch of I-80 in Northern California, with an on-ramp and an off-ramp. The highway has six lanes, one of which is a 'car-pool lane'. The data are broken up into three segments, each covering a fifteen minute period.

The data consists of detailed trajectory data for every car on the highway during the periods of observation. The trajectories are ultimately from multiple cameras installed along the highway. The advantage of this data set is that it provides a detailed 'ground truth' for the traffic, which is normally unknown.

From these trajectories, we extract data to use as the *sensor measurements* in our system, to represent loop detector measurements. These measurements are a subset of the information in the full trajectory data set. To create the measurements, we calculate density and velocity fields over each lane at each time step. The sensors measure these fields in the neighborhood around their fixed positions. For our experiments, we use only two sensors, each only viewing one *cell*-length (See Section 3.1.) of the flow field. The approximate locations of the sensors can be seen in Fig. 2. Ultimately the sensors each provide an average density and velocity reading for a time span, which corresponds to the kind of measurements actual loop detectors provide [Jia et al. 2001].

4.1 Metric

We present the results of our experiments using *lane-mean* values, defined for density as

$$\rho_l = \frac{\sum_{j=0}^{N-1} \rho_j}{N}. \tag{8}$$

This measure can be defined for the ground truth data (the hidden, full car trajectory data), the macroscopic state estimation, and the agent-based simulation. The velocity version of this calculation follows directly. This measure was used so as to demonstrate the global tracking ability of the system.

4.2 Highway Reconstruction Experiment

The results presented in Fig. 3 are for the time interval of 5:15 pm to 5:30 pm, when the highway traffic experiences congestion and traffic jams. In Figure 4, we see the lane-mean densities over time for a lane of reconstructed highway traffic. (Please see the Appendix in the supplementary document for other lanes). This scenario is fairly challenging as the traffic enters a congested state.

In these plots (shown in Fig. 4), we can see the state estimates of density and velocity for traffic along a highway lane (red) and the agent-based detailed reconstruction (blue) closely tracking the ground truth (green). The maximum error is on the order of 0.1 cars per 4.5 m (the car length) for the density tracking and 3 m/s for velocity tracking.

The root mean square (RMS) of the error for all lanes of the data set can be found in the Appendix. The density error ranges from 0.03 to 0.1 cars per car length (or proportion filled with cars), and the velocity error ranges from 1 to 2.5 meters per second.

4.3 Loop Detector Experiment

We have also reconstructed traffic flow from raw loop detector data from the PeMS system [Jia et al. 2001]. The state estimate and detailed reconstruction initially fail to track the sensors, but later converge. In Fig. 5, we can see that our method can track the mean of the loop detector measurements well once converged.[1]

For this experiment, we used two loop detector stations, with IDs 402615 and 403404, on one mile of I-880 northbound. We used the measurements from the second lane and reconstructed 3000 seconds of traffic flow, starting at 4:30 pm of 3/14/13. The interval between sensor measurements was 30 seconds.

4.4 Implementation Details

For our clover leaf scenario, seen in the suplimentary video, we used an ensemble size of 100. For our highway reconstruction, we increased the ensemble size to 250 to create a high quality reconstruction. The cell length, Δx, in each case was 9.4 meters. The timestep used for the ARZ simulator during the reconstruction was $\Delta t_1 = 0.3$. The timestep used for the detailed reconstruction was $\Delta t_2 = 0.05$ seconds. Sensor measurements occur at $\Delta t_3 = 15$ second intervals.

The motion model's $\mathbf{m}$ and observation model's $\mathbf{n}$ noise vectors were weighted with hand-tuned parameters. The parameter values were as follows. For the observation model, 0.15 was used for the density measurement, and 4.5 was used for the velocity measurement. For the motion model, 0.005 was used for the state density and relative velocity; 6 was used for the speed limit, 0.1 was used for γ, 0.2 was used for ramp density and relative velocity, and 0.3 was used for density and relative velocity at the lane boundaries.

4.5 Performance Analysis

Assuming the number of ensemble members dominates the dimension of the observation vector, the complexity of estimating the traffic state for one lane for one time interval is the cost of advancing the traffic states, $O(t_i m|\mathbf{x}|)$, plus the cost of the analysis operation, $O(t_i m|\mathbf{x}||\mathbf{z}|)$, where t_i is the number of sensor measurments, m is the number of ensemble members, $|\mathbf{x}|$ is the state size, $|\mathbf{z}|$ is the size of the observation vector, $f(\mathbf{x},\mathbf{m})$ is the cost of the motion model. The complexity of reconstruction for one lane for a timestep is $O(|\mathbf{x}|c)$, where c is the number of cars.

Our functional implementation serves more as a proof-of-concept than a benchmark; nonetheless, the efficiency of this technique is promising. We have tested a prototype implementation (single threaded, unvectorized code) on an Intel© Core™ i7-2820QM processor at 2.3 GHz. Our performance results are for a test scenario of a 1km road with 2 sensors. The ensemble size is set at 250, and the filter interval is set to 15 seconds. In total, 3000 seconds of traffic are reconstructed.

[1] Note that the loop-detector data has no values for the regions between sensors, so there is no true ground state with which to compare.

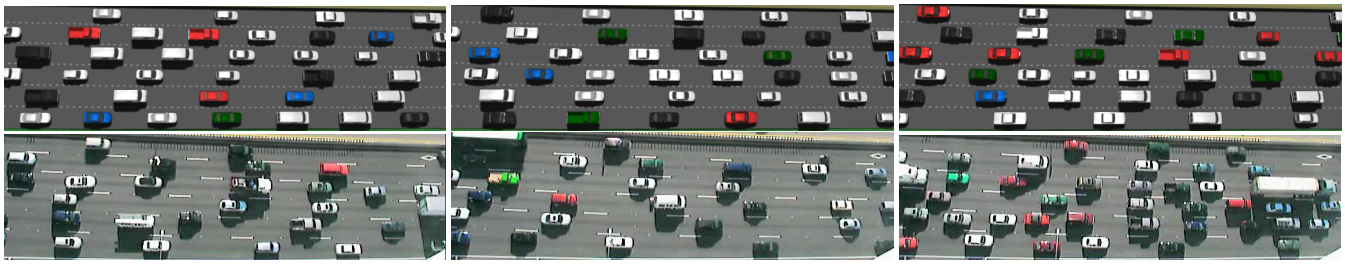

Figure 3: *In this figure, there are three comparisons between the reconstruction and video footage from 5:17, 5:21, 5:25 (Minutes two, six, and ten for the 5:15 data set). In the first image, we see our method over estimates the density in at least two of the lanes in this area. In the next two images, congested traffic is both reconstructed and found in the source data. We can see lane-specific features of the traffic are reconstructed: in all cases, the HOV lane shows a much lower density (and has a higher velocity) than the other lanes. (It should be noted that the camera placement for the reconstruction images is only approximate.)*

The run time is dominated by the state estimation, which takes 2.7 seconds per step, and 547.6 seconds total. This computation is almost completely dominated by a large ensemble of macroscopic traffic simulation runs, wherein each of the motion ensemble is simulated forward in time. The detailed reconstruction took 0.2 seconds per interval, and 42.1 seconds total.

The performance of this technique can be substantially accelerated through an architecturally-aware (such as GPU-based or many-core) implementation. In particular, there are multiple opportunities to exploit thread-level and data-level parallelism in the ensemble of many macroscopic simulations for state estimation. Using the increasing computational power will allow our method to effectively handle even larger road networks more efficiently.

While this work has focused on a single (multi-lane) stretch of road, our technique is easily made to scale to larger networks; multiple roads are loosely coupled at their endpoints, which ensures a high availability of thread-level parallelism. Furthermore, the algorithm is linear in each of the ensemble size and the number of cells – either of these can be refined and performance will vary linearly.

4.6 Comparison with Related Work

This is a similar problem to [van den Berg et al. 2009; Sewall et al. 2011a], however our approach is radically different and has multiple advantages. Unlike VT, which required individual car data and identification, we make realistic assumptions about sensor data. Furthermore, our method is real-time and scalable, whereas VT uses an exponentially complex per-car motion planning algorithm in a discretized acceleration space. Our method creates data-driven traffic animation from reconstructed flows and traffic models, allowing user-interaction and flexibility. Our work features a novel estimation approach using a state of the art simulation formulation, novel extensions to microscopic simulation, and efficient lane changing formulations, in addition to the generality of the overall approach. Our approach can lead to future work in user-controlled animation, traffic analysis and diagnosis, and numerous other extensions. Further details on specific differences follow.

- **Sensors:** Our work assumes a form of traffic sensor measurement that exists currently and is publicly available. We assume measurements of volume and velocity, which are available for in-road loop detectors [Rice and Van Zwet 2004] and can be derived from camera data. The earlier works [van den Berg et al. 2009; Sewall et al. 2011a] assumes readings for individual cars at road boundaries, which is not readily available. Furthermore, our work allows for sensors to be located arbitrarily along a lane.

- **Macroscopic:** The related work reconstruct trajectories for individual cars. Our approach focuses on a macroscopic reconstruction first, and then a visualization consisting of individual cars. This has two benefits. First, it protects the privacy of drivers by making it impossible to trace individual cars. Second, it drastically decreases the amount of information needed to perform a reconstruction. In our work, the size of the sensor data is constant, while in the previous work the size scales with the number of cars.

- **Simulation:** Our work creates a simulation-based reconstruction with which a user could interact. A user could reconstruct and visualize traffic, then drive a virtual car in that traffic. The reconstructed cars would react to the user, for example allowing the user to create a traffic jam when one did not occur in the observed data. This capability is not possible with the earlier approach.

In terms of quantitative accuracy, it is difficult to make a comparison between the two approaches due to their differing formulations. However, in [Sewall et al. 2011a], the authors found that their method was able to find trajectories for 82% of the vehicles. This statistic may be viewed roughly as a measure of density accuracy. However, this metric is simplistic for a few reasons. First, it ignores the variation between lanes. Some lanes may be congested while others are free flowing. Second, it ignores inner lane density variations, i.e. a traffic jam on the first part of the highway and free flowing traffic farther downstream. However, ignoring these complications, we can state that our method can reconstruct the motion of all vehicles and it should have at least comparable density accuracy, if not higher at several places(see Appendix for details). It is not possible to compare the accuracy in the velocity field reconstructions with earlier methods, as such information was not available.

Our work compares favorably with the prior work in terms of computational cost, where [Sewall et al. 2011a] reconstructed 15 minutes of real-world traffic in 6.64 minutes of computation. Using the same dataset, our method can reconstruct 15 minutes of traffic in a matter of seconds, as discussed above in Sec. 4.5. This performance improvement can be easily explained by the fact that our method does not require an expensive, priority-based, multi-agent route-planning algorithm.

5 Conclusion

We have presented a real-time traffic reconstruction and visualization algorithm using traffic sensor data obtained from in-road loop detectors. Our approach seamlessly integrates an efficient state estimation method using Ensemble Kalman Filter and continuum traffic simulation with a fast traffic reconstruction using an agent-based traffic simulation system that produces realistic motion for individual vehicles whose global states (e.g. density and velocity fields) are consistent with the estimated traffic flow in the real world and

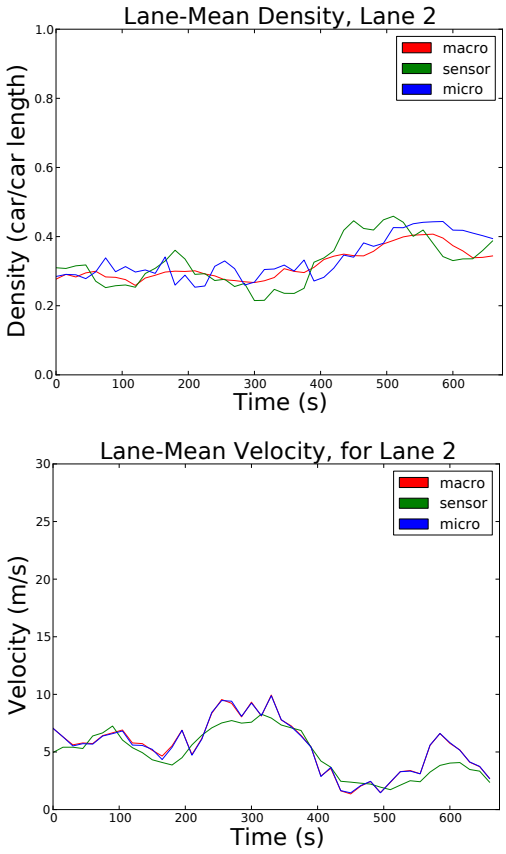

Figure 4: *The lane-mean density and velocity for a lane of highway traffic. The* **green** *line is the ground truth, the* **red** *line is the state estimate, and the* **blue** *line is the agent-based simulation.*

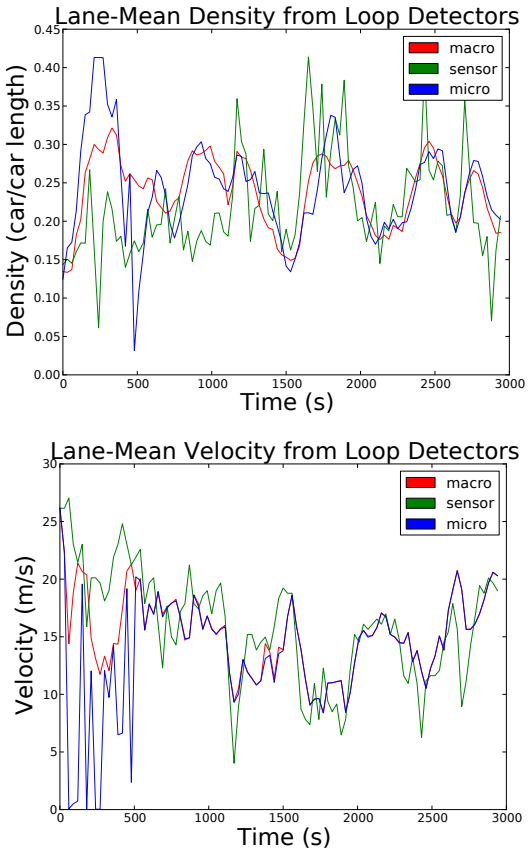

Figure 5: *The lane-mean density and velocity for a lane of highway traffic reconstructed from loop detectors.*

individual vehicle's kinematic and dynamic constraints. The computational framework presented here has been implemented, tested, and validated on real-world traffic data.

5.1 Limitations

Complex traffic phenomena happen primarily on highways, therefore we focus on this aspect. Intersections are a difficult issue in regard to data-driven animation as they have their own states. However, this work could be easily combined with a microscopic simulation of an urban area with intersections.

The accuracy of our method is limited by the available data. High frequency traffic phenomena can be missed if the data is too temporally sparse. Our estimation method is fundamentally macroscopic, so one should not expect individual cars to match.

In certain scenarios, the macroscopic estimate and the microscopic detailed reconstruction could diverge as our method does not guarantee vehicles are conserved in the macroscopic estimate. In some cases, this could be handled by animators. The excess or deficit microscopic flow can be corrected by adding or removing cars at acceptable points, such as ramps, off screen, etc. However, these boundary points may not always be present. Further, if the boundary points themselves have sensors and estimates, the addition/removal of flow could cause the microscopic state of the boundary points (e.g. highway ramps) to diverge. This could create an incorrect visualization of cars entering or leaving the network at locations inconsistent with the real world.

5.2 Future Work

We would like to extend this work to handle large, complete networks of highways and the road networks surrounding them. Another interesting extension of this work would be to add visualization of traffic predictions along with the current state estimate. We can also use both historical data, as well as driver specific data and cell-phone sensing to improve the filtering performance. Finally, our system could be combined with routing algorithms to provide both efficient plans and visualizations of those routes on virtual globe systems.

Acknowledgement:

We would like to thank Sujeong Kim for video editing and for comments and feedback. We would like to thank Jur van den Berg for suggestions and early discussion on this project. This research was supported in part by National Science Foundation, Army Research Office, Intel, and Carolina Development Foundation.

References

AW, A., AND RASCLE, M. 2000. Resurrection of" second order" models of traffic flow. *SIAM journal on applied mathematics*, 916–938.

CREMER, J., KEARNEY, J., AND WILLEMSEN, P. 1997. Directable behavior models for virtual driving scenarios. *Trans. Soc. Comput. Simul. Int. 14*, 2, 87–96.

CREMER, M. 1991. Flow variables: estimation. *Concise encyclopedia of traffic and transportation systems*, 143–148.

DONIKIAN, S., MOREAU, G., AND THOMAS, G. 1999. Multimodal driving simulation in realistic urban environments. *Progress in System and Robot Analysis and Control Design (LNCIS) 243*, 321–332.

EVENSEN, G. 2003. The ensemble kalman filter: Theoretical formulation and practical implementation. *Ocean dynamics 53*, 4, 343–367.

HEGYI, A., MIHAYLOVA, L., BOEL, R., AND LENDEK, Z. 2007. Parallelized particle filtering for freeway traffic state tracking. In *Proc. European Control Conference*.

HELBING, D. 2001. Traffic and related self-driven many-particle systems. *Reviews of modern physics 73*, 4, 1067.

HOUTEKAMER, P., AND MITCHELL, H. 2001. A sequential ensemble kalman filter for atmospheric data assimilation. *Monthly Weather Review 129*, 2, 123–137.

JACQUET, D., CANUDAS DE WIT, C., AND KOENIG, D. 2005. Traffic control and monitoring with a macroscopic model in the presence of strong congestion waves. In *Proc. Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, IEEE, 2164–2169.

JACQUET, D., KRSTIC, M., AND DE WIT, C. 2006. Optimal control of scalar one-dimensional conservation laws. In *Proc. American Control Conference, 2006*, IEEE, 6–pp.

JIA, Z., CHEN, C., COIFMAN, B., AND VARAIYA, P. 2001. The PeMS algorithms for accurate, real-time estimates of g-factors and speeds from single-loop detectors. In *Proc. IEEE Intelligent Transportation Systems*, IEEE, 536–541.

LARAMEE, R. S., HAUSER, H., DOLEISCH, H., VROLIJK, B., POST, F. H., AND WEISKOPF, D. 2004. The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum 23*, 2, 203–221.

LEBACQUE, J.-P., MAMMAR, S., AND HAJ-SALEM, H. 2007. The aw-rascle and zhang's model: Vacuum problems, existence and regularity of the solutions of the riemann problem. *Transportation Research Part B*, 41, 710–721.

LIGHTHILL, M., AND WHITHAM, G. 1955. On kinematic waves. ii. a theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences 229*, 1178, 317–345.

2011. MIT Intelligent Transportation Systems. `http://mit.edu/its/mitsimlab.html`.

2013. Next Generation SIMulation. `http://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm`.

PAPAGEORGIOU, M., BLOSSEVILLE, J., AND HADJ-SALEM, H. 1990. Modelling and real-time control of traffic flow on the southern part of boulevard périphérique in paris: Part i: Modelling. *Transportation Research Part A: General 24*, 5, 345–359.

PAUSCH, R., CREA, T., AND CONWAY, M. 1992. A literature survey for virtual environments - military flight simulator visual systems and simulator sickness. *Presence: Teleoperators and Virtual Environments 1*, 3, 344–363.

RICE, J., AND VAN ZWET, E. 2004. A simple and effective method for predicting travel times on freeways. *Intelligent Transportation Systems, IEEE Transactions on 5*, 3, 200–207.

RICHARDS, P. 1956. Shock waves on the highway. *Operations research*, 42–51.

SAU, J., EL FAOUZI, N., AISSA, A., AND DE MOUZON, O. 2007. Particle filter-based real-time estimation and prediction of traffic conditions. *Recent advances in stochastic modeling and data analysis: Chania, Greece, 29 May-1 June 2007*, 406.

SEWALL, J., WILKIE, D., MERRELL, P., AND LIN, M. 2010. Continuum traffic simulation. In *Computer Graphics Forum*, vol. 29, Wiley Online Library, 439–448.

SEWALL, J., VAN DEN BERG, J., LIN, M., AND MANOCHA, D. 2011. Virtualized traffic: Reconstructing traffic flows from discrete spatiotemporal data. *Visualization and Computer Graphics, IEEE Transactions on 17*, 1, 26–37.

SEWALL, J., WILKIE, D., AND LIN, M. C. 2011. Interactive hybrid simulation of large-scale traffic. *ACM Transaction on Graphics (Proceedings of SIGGRAPH Asia) 30*, 6 (December).

SEWALL, J. 2011. *Efficient, Scalable Traffic and Compressible Fluid Simulations Using Hyperbolic Models*. PhD thesis, University of North Carolina at Chapel Hill.

THOMAS, G., AND DONIKIAN, S. 2000. Modelling virtual cities dedicated to behavioural animation. *Computer Graphics Forum 19*, 3.

TREIBER, M., HENNECKE, A., AND HELBING, D. 2000. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E 62*, 2, 1805–1824.

VAN DEN BERG, J., SEWALL, J., LIN, M., AND MANOCHA, D. 2009. Virtualized traffic: Reconstructing traffic flows from discrete spatio-temporal data. In *Proc. IEEE Virtual Reality Conference.*, IEEE, 183–190.

WANG, Y., AND PAPAGEORGIOU, M. 2005. Real-time freeway traffic state estimation based on extended kalman filter: a general approach. *Transportation Research Part B: Methodological 39*, 2, 141–167.

WANG, H., KEARNEY, J., CREMER, J., AND WILLEMSEN, P. 2005. Steering behaviors for autonomous vehicles in virtual environments. In *Proc. IEEE Virtual Reality Conf.*, 155–162.

WILKIE, D., VAN DEN BERG, J., LIN, M., AND MANOCHA, D. 2011. Self-Aware traffic route planning. *Proc. of AAAI 2011*.

WILKIE, D., SEWALL, J., AND LIN, M. 2012. Transforming GIS data into functional road models for large-scale traffic simulation. *IEEE Trans. on Visualization and Computer Graphics 18*, 6.

WORK, D., TOSSAVAINEN, O., BLANDIN, S., BAYEN, A., IWUCHUKWU, T., AND TRACTON, K. 2008. An ensemble kalman filtering approach to highway traffic estimation using gps enabled mobile devices. In *Proc. IEEE Decision and Control*, IEEE, 5062–5068.

WORK, D., BLANDIN, S., TOSSAVAINEN, O., PICCOLI, B., AND BAYEN, A. 2010. A traffic model for velocity data assimilation. *Applied Mathematics Research eXpress*.

ZHANG, H. 2002. A non-equilibrium traffic model devoid of gas-like behavior. *Transportation Research Part B: Methodological 36*, 3, 275–290.