2025 IEEE 64th Conference on Decision and Control (CDC)
December 10-12, 2025. Rio de Janeiro, Brazil

Online Traffic Density Estimation using Physics-Informed Neural Networks

Dennis Wilkman[1] and Kateryna Morozovska[1] and Karl Henrik Johansson[1] and Matthieu Barreau[1]

2025 IEEE 64th Conference on Decision and Control (CDC) | 979-8-3315-2627-6/25/$31.00 ©2025 IEEE | DOI: 10.1109/CDC57313.2025.11312033

***Abstract*— Recent works on applying Physics-Informed Neural Networks to traffic density estimation have shown promise for future developments due to their robustness to model errors and noisy data. In this paper, we introduce a methodology for online approximation of the traffic density using measurements from probe vehicles in two settings: one using the Greenshield model and the other considering a high-fidelity traffic simulation. The proposed method continuously estimates the real-time traffic density in space and performs model identification with each new set of measurements. The density estimate is updated in almost real-time using gradient descent and adaptive weights. In the case of full model knowledge, the resulting algorithm performs similarly to the classical open-loop one. However, in the case of model mismatch, the iterative solution behaves as a closed-loop observer and outperforms the baseline method. Similarly, the proposed algorithm correctly reproduces the traffic characteristics in the high-fidelity setting.**

I. Introduction

The transport system accounts for approximately 18% of global energy consumption [1]. Optimizing traffic flow can be one of the solutions to meet the UN's Sustainable Development Goal 11 by promoting sustainable and efficient urban mobility. The possibility of using rapid developments in autonomous vehicles for traffic state control reduces emissions while keeping travel time low by resolving traffic jams and stabilizing human-controlled vehicles [2], [3].

Like many control strategies, effective autonomous transport requires an accurate traffic density estimation. Traditional methods require sensor-based monitoring fixed on the road or camera surveillance. They depend on the surrounding infrastructure, thus suffering from high costs, limited scalability, and real-time adaptability. Instead, data from probing vehicles traversing the road can be collected to avoid using fixed sensors. However, this new trend relies on a high share of cars with probing capabilities. A solution is to consider mixed traffic and model traffic as a continuous 'fluid' instead of individual vehicles, which can maximize the benefits of mobile sensors [4]. Thus, traffic density can be estimated using the first-order Lighthill-Withham-Richards (LWR) model [5], [6], by using hyperbolic partial differential equations (PDEs) to describe traffic flow. For better representation of the real measurements, higher-order models have been developed in the high-fidelity simulation software SUMO [7].

*This work is supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

[1]Authors are with the Division of Decision and Control Systems, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology, 100-44, Sweden {dwilkman, kmor, kalle, barreau}@kth.se

Physics-informed neural networks (PINNs) have emerged as a promising approach for solving merging model and data-based solutions. PINNs are trained to solve supervised learning tasks while adhering to the physical laws of the given system [8]. They can efficiently model traffic flow by combining the LWR model [9] and density measurements sampled from probe vehicles [10], [11]. PINNs are also more effective for traffic state estimation than the traditional control-based approaches, such as Extended Kalman Filters [12] or open-loop observers [13] when handling high-fidelity, sparse, and noisy offline data.

In control applications, density estimation should have high accuracy and robustness and be achieved in almost real-time. Therefore, we propose a framework for real-time density estimation by adapting PINNs for online training. The resulting algorithm shows the benefits of PINNs applied to the traffic density problem, while considering various cases such as the required number of past data points to store, training time, and its effects on the performance of the estimation. We also investigate its performance in the case of model mismatch on a high-fidelity SUMO simulation.

This manuscript is structured as follows. Section II contains the background on macroscopic traffic flow modeling, PINNs, and the problem statement. Section III presents a methodology for training and evaluating the online density estimation models, and outlines the assumptions considered for model training. Section IV presents the model implementation results in two settings: the Greenshield model and high-fidelity simulations. When appropriate, a comparison between the PINN model and an open-loop observer is made. Finally, Section V summarizes the conclusions and outlines potential future research directions.

II. Background and Problem Statement

This section describes the macroscopic and microscopic models used for modeling the dynamics of the traffic flow, as well as gives an introduction to PINNs implementation for traffic density reconstruction.

A. Macroscopic and microscopic modeling of traffic flow

Traffic flow can be modeled using different approaches, either with a microscopic traffic model by considering behaviors and interactions between the individual vehicles or with a macroscopic traffic model by disregarding the individual behaviors and focusing on the large-scale traffic properties.

Traffic flow modeled using LWR can be expressed using the continuity equation on the domain $\Omega = [0, T] \times \mathbb{R}$:

$$\partial_t \rho + \partial_x (\rho v) = 0, \tag{1}$$

where ∂_t and ∂_x are the partial derivatives with respect to t and x, respectively. The density $\rho(t,x)$ is the 'fluid' density or the number of vehicles at location x and time t, $\rho \in [0, \rho_{max}]$. The maximum density ρ_{max} is equal to the inverse of the average length of a vehicle plus the average distance between vehicles during a traffic jam. The velocity $v(\rho)$ is the instantaneous microscopic speed given density ρ.

The velocity equals the free flow velocity at 0 density, $v(0) = v_f$. It is a decreasing function, such that at the maximum density ρ_{max}, it should be equal to zero $v(\rho_{max}) = 0$ [10]. This means that $\rho v(\rho)$ is a concave function that ensures the existence of a solution [14, Theorem 6.2.2] to the system (1). A straightforward model for the microscopic velocity is given as $v = v_f \left(1 - \frac{\rho}{\rho_{max}}\right)$ [15].

Since (1) does not have a unique solution and it might be discontinuous, it is common to consider the viscous form of (1) instead. The viscous equation includes the diffusion term $\gamma \partial_{xx}\rho$ where $0 < \gamma \ll 1$, ensuring the numerical stability and uniqueness of the solution, together with smoother properties [16]. The resulting macroscopic model with normalized density $\rho \in [0,1]$ on the same domain $\Omega = [0,T] \times \mathbb{R}$ writes as

$$\begin{cases} \partial_t \rho + v_f(1-2\rho)\partial_x \rho = \gamma \partial_{xx}\rho, \\ \rho(0,\cdot) = \rho_0. \end{cases} \tag{2}$$

From a microscopic perspective, the penetration rate defines the fraction of available individual vehicles on the road equipped with probing sensors. If vehicle i is one of them, then its position x_i follows the dynamics equation

$$\dot{x}_i(t) = v_i(t, x_i) = v(\rho(t, x_i(t))). \tag{3}$$

Since both the micro and macro formulations are using the same velocity function $v(\rho)$ [16], the coupled micro-macro traffic model can be used to connect the behaviors of individual probe vehicles to the large-scale property density of the traffic across the entire road:

$$\begin{cases} \partial_t \rho(t,x) = -v_f(1-2\rho)\partial_x \rho + \gamma \partial_{xx}\rho, \\ \dot{x}_i(t) = v(\rho(t, x_i(t)). \end{cases} \tag{4}$$

In Fig. 1 we see the simulated solution and the paths of the probe vehicles through space and time with the Greenshield closure equation $v(\rho) = v_f(1-\rho)$.

B. Physics Informed Neural Network

A feedforward neural network is defined as

$$\mathcal{N}[\theta](\nu) = W_L H_{L-1} \circ H_{L-2} \circ \cdots H_1(\nu) + b_L, \tag{5}$$

where $\theta = \{(W_i, b_i)\}_{i=1,\dots,L}$ is the tensor of parameters, $W_i \in \mathbb{R}^{N \times N}$ is the weight, $b_i \in \mathbb{R}^{N \times 1}$ is the bias, N is the number of neurons, $H_i(\nu) = \phi(W_i \nu + b_i)$ is the layer i, ϕ is an element-wise activation function and L is the number of layers. The first and last layers have appropriate dimensions according to the input and output dimensions of the network. The universal approximation theorem [17] states that a feedforward neural network with $L > 0$ can approximate arbitrarily close any smooth function, provided a sufficient number of neurons N.

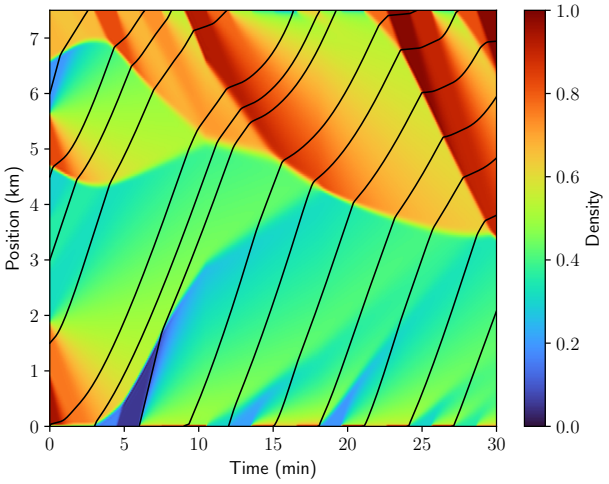

Fig. 1. The black lines are the paths x_i of the probe vehicles i, in which sensor data $\rho(\cdot, x_i(\cdot))$ is measured.

The PINN is trained to approximate a solution and/or parameters of the PDE from the data. PINN is defined as the optimization under constraints problem

$$\begin{aligned} \theta^* = \ & \text{Argmin}_\theta \quad \frac{1}{|\mathcal{D}|} \sum_{(\nu,y)\in\mathcal{D}} \|y - \mathcal{N}[\theta](\nu)\|^2 \\ & \text{s.t.} \quad \int_\Omega \|\mathfrak{F}\left[\mathcal{N}[\theta]\right](\nu)\|^2 d\nu = 0, \end{aligned} \tag{6}$$

where the dynamical constraints, expressed as differential equations, are defined as $\mathfrak{F}[\mathcal{N}[\theta]] = 0$ on Ω. The measurements are formulated as a dataset $\mathcal{D} = \{(\nu_i, y_i)\}_i$ where ν_i is the input giving the output y_i.

Problem (6), however, is not numerically tractable. A solution is to relax it, leading to:

$$\begin{aligned} \min_\theta \ \max_\Lambda \frac{1}{|\mathcal{D}|} \sum_{(\nu,y)\in\mathcal{D}} & \|y - \mathcal{N}[\theta](\nu)\|^2 \\ & + \frac{1}{|\mathcal{D}_\mathfrak{F}|} \sum_{\nu \in \mathcal{D}_\mathfrak{F}} \left\|\mathfrak{F}\left[\mathcal{N}[\theta]\right](\nu)\right\|_\Lambda^2, \end{aligned} \tag{7}$$

where $\mathcal{D}_\mathfrak{F} \subset \Omega$ is a uniform sampling over Ω and $\|A\|_\Lambda^2 = A^\top \Lambda A$ for Λ definite positive. In that formulation, Λ is a positive diagonal matrix representing the Lagrange multipliers and is used to penalize divergence of the physics residuals. Problem (7) is a learning problem that can be solved using gradient-descent on θ and gradient-ascent on Λ [18].

C. PINNs and traffic state reconstruction

Based on the original work in [10], the PINN model for traffic density estimation can be defined as (6) with

$$\hat{\rho}[\theta](t,x) = \mathcal{N}[\theta]\left([t,x]\right), \tag{8}$$

$$\hat{v}[\psi](\rho) = (1-\rho)\left(v_f + \rho \mathcal{N}[\psi](\rho)^2\right), \tag{9}$$

$$\mathfrak{F}[\hat{\rho}, \hat{v}] = \begin{bmatrix} \partial_t \hat{\rho} + (\hat{v} + \rho \hat{v}')\,\partial_x \hat{\rho} - \gamma \partial_{xx}\hat{\rho} \\ \max(\hat{v}', 0) \end{bmatrix}, \tag{10}$$

with $\hat{v}' = \frac{d\hat{v}}{d\rho}$ and $(t,x) \in \Omega = [0,T] \times [0,L]$.

The dataset $\mathcal{D} = \mathcal{D}(0,T)$ is defined as:

$$\mathcal{D}(t_1,t_2) = \bigcup_{i\in\{1,\dots,N\}} \mathcal{D}_i(t_1,t_2), \tag{11}$$

where

$$\mathcal{D}_i(t_1,t_2) = \{(t_k, x_i(t_k), v_i(t_k), \rho(t_k, x_i(t_k))\}_{t_k\in[t_1,t_2]}, \tag{12}$$

where $\{t_k\}$ are the sampling times and $v_i(t_k)$ is the speed of probe vehicle i at time t_k. The set $\mathcal{D}_i(t_1,t_2)$ refers to the data collected by probe vehicle i between time t_1 and t_2. The data loss function can then be expressed as

$$\mathcal{L}_\rho(t_1,t_2) = \sum_{i=1}^{N} \sum_{(t,x,v,\rho)\in\mathcal{D}_i(t_1,t_2)} \Big\{ \big|\hat{\rho}[\theta](t,x) - \rho\big|^2 + \big|\hat{v}[\psi](\rho) - v\big|^2 \Big\}, \tag{13}$$

while the physics loss function is defined as:

$$\mathcal{L}_\phi(t_1,t_2,\Lambda) = \frac{1}{|\mathcal{D}_{\mathfrak{F}}(t_1,t_2)|} \sum_{\nu\in\mathcal{D}_{\mathfrak{F}}(t_1,t_2)} \big\|\mathfrak{F}\big[\mathcal{N}[\theta]\big](\nu)\big\|_\Lambda^2, \tag{14}$$

where $\mathcal{D}_{\mathfrak{F}}(t_1,t_2)$ is a uniform sampling of collocation points over $[t_1,t_2]\times[0,L]$. In Fig. 2, we illustrate the general training method in this setting, formulated by the following optimization problem:

$$\theta^*(t_1,t_2,\delta) = \underset{\theta}{\text{Argmin}} \max_{\Lambda} \mathcal{L}_\rho(t_1,t_2) + \mathcal{L}_\phi(t_1,t_2+\delta,\Lambda) \tag{15}$$

and the optimal reconstruction is $\hat{\rho}[\theta^*(0,T,\delta)]$.

Note that this approach is particularly well-suited to ill-posed problems where there are not enough or too many measurements to ensure the existence of a unique solution. In our case, probe vehicle measurements introduce redundancy and then possibly over-constrain the problem. At the same time, this redundancy can be exploited to uncover some unknown parts of the model dynamics, making this framework adaptable to model identification [10].

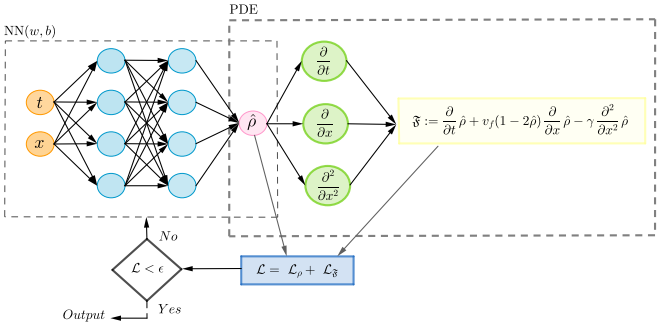

Fig. 2. The PINN for traffic density reconstruction.

Another important aspect of this method is the extension to prediction. In that case, future values of the density are estimated using the model $\mathfrak{F}[\hat{\rho},\hat{v}] = 0$ over a larger domain $[0, T+\Delta T]\times[0,L]$. Therefore, the neural network

$$\hat{\rho}_T = \hat{\rho}\big[\theta^*(0,T,\Delta T)\big] \tag{16}$$

can be meaningfully evaluated on $[0, T+\Delta T]$ even if no measurements are collected during $[T, T+\Delta T]$.

D. Problem Statement

The traffic state reconstruction approach shows excellent estimation capabilities, with model identification and prediction included in the methodology. However, approximating the solution using gradient descent is a very slow and computationally demanding process. This leads to longer training times, which prevents its use for real-time problems.

The first objective is to find a causal algorithm that continuously gives an accurate estimate of the real-time traffic density. The new sensor measurements are sampled in real-time from probe vehicles. We are interested in minimizing the Current Estimation Error (CEE) between the true density $\rho(t,\cdot)$ and its estimation $\hat{\rho}(t,\cdot)$, given as:

$$CEE_t(\rho,\hat{\rho}) = \int_0^L |\rho(t,x) - \hat{\rho}(t,x)|^2 dx. \tag{17}$$

Note that causality implies that, at time t, the estimation $\hat{\rho}$ only uses the data available until the same time t.

Another objective is to keep the CEE low even in case of model mismatch, a difference between the assumed and actual dynamics of the system. The proposed method should be able to adapt in real-time to reduce its impact on estimation errors.

III. Methodology

This section covers a general framework for real-time density estimation with PINNs and discusses the fundamental limitations and how training can be adapted to that purpose.

A. Framework for online real-time estimation through iterative learning

The main idea behind this framework is to leverage the predictive capabilities of PINNs as a buffer to perform offline operations.

At time $T > 0$, we have data obtained from probing vehicles in the time window $[0,T]$. If we begin training the PINNs at time T on the domain $[0, T+\Delta T]$, we get $\hat{\rho}_T$ after solving the optimization problem (15). Note that ΔT is a parameter to be chosen. Since training and inferring from the neural network is a slow process, it will take δ_t seconds, and the estimate will be ready at time $T+\delta_t$.

Therefore, from time $T+\delta_t$, we will have the predictions trained on the data between time $[0,T]$. If we train a new model continuously, then at time $T+\delta_t$, we will begin training a new model for which inference will be available at time $T+2\delta_t$. Consequently, to have an estimate at any time between $[T, T+2\delta_t]$, the first training should include prediction over $[T, T+2\delta_t]$, meaning that $\Delta T = 2\delta_t$. In that case, we can use equation (16) and $\hat{\rho}_{\delta_t}$ will produce the estimate for $t\in[T+\delta_t, T+2\delta_t]$. Pursuing this logic and assuming that $T = \delta_t$, we will use the following reconstructed state for any t larger than $2\delta_t$:

$$\hat{\rho}(t,\cdot) = \begin{cases} \hat{\rho}_{\delta_t}(t,\cdot) & \text{if } t\in[2\delta_t, 3\delta_t), \\ \vdots & \\ \hat{\rho}_{i\delta_t}(t,\cdot) & \text{if } t\in[(i+1)\delta_t, (i+2)\delta_t), \\ \vdots & \end{cases} \tag{18}$$

Fig. 3 illustrates the process of continuously training models using the proposed framework.

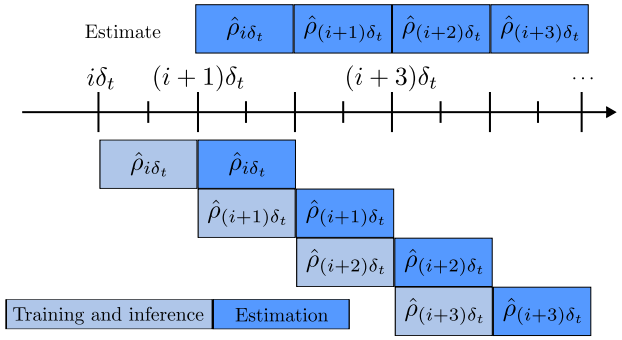

Fig. 3. Framework for using a PINN for online real-time estimation using iterative learning.

B. Practical considerations

The previous subsection introduced a framework for using PINNs in a real-time setting. However, certain limitations are also addressed in this subsection.

1) Infinite horizon training: In the given framework, the size of the inference domain continues to increase with each iteration. At time $i\delta_t$ with $i > 2$, the size of the PINN domain is equal to $(i+2)\delta_t L$. As defined in [18, Proposition 1], the accuracy of a neural network is, at worst, proportional to the size of the inference window. This implies a larger neural network to enable a low L^2 error over the domain, leading to the first research question.

Question 1: How to keep the size of the neural network constant with a constant accuracy?

2) Training Time: For real-time estimations, a significant challenge is the impact of the training time on the quality of the predictions. Hence, if the model has completed training and inferred at time t, it will only have been trained on data up to $t-\delta_t$. The longer the training time δ_t, the less relevant the data in the training set becomes. Therefore, computationally heavy methods that traditionally are expected to improve the accuracy of the estimation may instead have the opposite effect due to the increased training time δ_t.

For the proposed framework, the increasing domain size will increase the number of training iterations (larger neural network) and the number of data points. Assume that the training time δ_t at time T is approximately equal to $e_t(\alpha|\mathcal{D}_{\mathfrak{F}}(0, T+2\delta_t)| + \beta|\mathcal{D}(0,T)|)$, where e_t is the number of training iterations, and α, β are determined by factors such as network architecture and training setup. Then, δ_t will be unbounded and, depending on the number of physics collocation points used, the effects on the training time from the number of data points in the data set $|\mathcal{D}_{\mathfrak{F}}(t_1, t_2)|$ can vary substantially. This leads to the following research question.

Question 2: How to ensure the accuracy of PINNs with constant training time?

3) Robustness: Traffic is often stochastic. With the LWR framework, the free flow velocity v_f might change with time and space. This would reduce the estimation quality due to a mismatch between the real and assumed model. The same applies to sensor noise, which can be affected by the current conditions. This leads to the following question.

Question 3: How to keep the model robust against model errors in an online setting?

C. Extending PINNs for online learning

We introduce adaptations of the classical PINNs training to extend to practical online learning by proposing answers to the previously mentioned questions.

1) Training time-window: The size of the dataset $\mathcal{D}$ grows with time, which eventually causes the training time δ_t to become unreasonably large, and the increasing size of the training domain causes the neural network to produce worse results. The simplest solution is to introduce a maximum look-back time δ_d. This modification creates a moving-time window $[T-\delta_d, T]$ for storing past data. Using formulation (16), we introduce:

$$\hat{\rho}_{T,\delta_d} = \hat{\rho}\big[\theta^*(T-\delta_d, T, 2\delta_t)\big]. \tag{19}$$

Therefore, if a model is continuously trained, the new reconstructed density is:

$$\hat{\rho}(t,\cdot) = \hat{\rho}_{i\delta_t,\delta_d}(t,\cdot), \tag{20}$$

where $i = \lfloor t/\delta_t \rfloor - 1$.

This approach implies that at time T, we forget about data gathered before $T-\delta_d$. Based on [19], there always is a time window δ_d such that (4) has a unique solution after time T using measurements from probing vehicles as boundary conditions. However, if the densities are small, δ_d can be arbitrarily large. Consequently, the time window introduces a trade-off between the uncertainty in the reconstruction at low densities, the size of the neural network, and the training time.

2) Efficient training of PINNs in an online setting: Here, we consider using the time window defined above. At the beginning of the ith training, starting at time $i\delta_t$, consider the time window of the data to be $\mathcal{I}_i = [i\delta_t - \delta_d, i\delta_t]$ with $\delta_d > \delta_t$, to ensure no times exist where no data is collected. The optimization (15) is then conducted from scratch, using, for instance, Xavier initialization for the parameters θ. This is a waste of computational resources since the model trained at time $(i-1)\delta_t$ considered the time-window $\mathcal{I}_{i-1}$ and $\mathcal{I}_{i-1} \cap \mathcal{I}_i$ is not empty. Consequently, the previous model contains valuable information that can be transferred to the new model. This could serve both as a warm startup, which will significantly decrease the training time, and as a propagation of the initial condition from the last time step.

Reusing a previous model for the new training is possible by normalizing and shifting it. First, since the time input for a given model i belongs to the window $[i\delta_t - \delta_d, (i+2)\delta_t]$, one can define the transformation:

$$s^{(i)} : t \mapsto \frac{2}{2\delta_t + \delta_d}\left(t - (i+1)\delta_t + \frac{\delta_d}{2}\right), \tag{21}$$

which will ensure that $s^{(i)}(t) \in [-1, 1]$ if $t \in [i\delta_t - \delta_d, (i+2)\delta_t]$. This normalization is a pre-processing step

to guarantee that the input is bounded. Then, it is possible to keep almost all the parameters from the previous training, up to a change in the bias b_1 to introduce a time shift:

$$\hat{\rho}_{i\delta_t,\delta_d}\left((i+2)\delta_t,\cdot\right)=\hat{\rho}_{(i+1)\delta_t,\delta_d}\left((i+2)\delta_t,\cdot\right). \tag{22}$$

If the bias of the first layer for the iteration i is denoted by $b_1^{(i)}$ given $\mathbb{1}=\begin{bmatrix}1 & 0\end{bmatrix}$, then the previous equality implies:

$$\mathbb{1}W_1 s^{(i)}((i+2)\delta_t)+\mathbb{1}b_1^{(i)}=\mathbb{1}W_1 s^{(i+1)}((i+2)\delta_t)+\mathbb{1}b_1^{(i+1)},$$

for any $x\in[0,L]$, which leads to:

$$b_1^{(i+1)}=b_1^{(i)}+\begin{bmatrix}\mathbb{1}W_1\left[s^{(i)}((i+2)\delta_t)-s^{(i+1)}((i+2)\delta_t)\right]\\ 0\end{bmatrix}.$$

By using this transformation before the next iteration of training, the information from before $i\delta_t-\delta_d$ is also preserved as a warm startup, allowing for a potentially greater information retention of data before the window size δ_d.

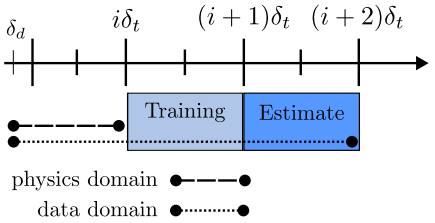

Fig. 4. The time scale for training a new PINN iteration at time $i\delta_t$.

3) Model identification capabilities in an online setting: In the online case, it is reasonable to assume that v_f can vary with time, due to various factors such as road accidents or weather conditions. Since an estimate of the free flow velocity is required for our framework, we need to learn it in real-time. However, it is not physically accepted that v_f varies at each training iteration. Therefore, we propose considering the window size δ_v for fitting velocity measurements based on two considerations: a short window allows for faster adaptation, while a longer one increases the size of the dataset. The latter improves robustness to noisy measurements of traffic densities and velocities due to increased redundancy in the data samples. Additionally, a broader range of densities allows the PINN to learn more complex traffic models and functional relationships between ρ and v.

IV. Results

The proposed PINN framework applied to the Greenshield model is compared with a traditional open-loop observer for different degrees of model mismatch, and we investigate the impact of training time on the error. Later, the results also discuss the model's performance with SUMO.

A. Greenshield model

The time windows of the data and velocity measurements are set to $\delta_d=\delta_v=3min$, the sampling rate f_s is set to 3 samples per second, the number of training iterations e_t is set to 100 epochs, and the neural network has $L=2$ layers and $N=32$ neurons making $\delta_t=0.3min$. Note that δ_t is dependent on the available computing power.

The simulation data is generated by solving (2) using a finite difference method [20] with a maximum value of $v_f=37.5km/h$, and the boundary conditions of ρ as piecewise constants in the range $[0,1]$. The free flow velocity varies with time such that:

$$v_f(t)=\begin{cases}37.5km/h & \text{if } t\in[0,10),\\ 18.75km/h & \text{if } t\in[10,18),\\ 30km/h & \text{if } t\in[18,30].\end{cases}$$

For the real-time PINN, the free flow velocity v_f is learned from data, and the updates are sufficiently quick so that δ_t is assumed to equal the sampling rate. We compare our results with the open-loop observer designed in [13] with specified $v_f=37.5km/h$.

1) Comparison between open-loop observer and PINN with varying mismatches in model knowledge: Fig. 5 shows the results with a varying free flow velocity, for the observer and the respective real-time estimates of the observer and the PINN. Fig. 6 illustrates the resulting CEE across time for the observer and PINN. In the initial time between $t\in[0,7]$, when the traffic density is higher, we can see how the observer error is decreasing rapidly. In contrast, the PINN error fluctuates, most likely due to the lack of training iterations required for convergence. Between $t\in[7,10]$, we can see the error spikes and increases for the observer as the boundary condition changes to low-density flow. In the interval of the perfect model knowledge $t\in[0,10]$, the observer outperforms the PINN model. The lower density also affects the PINN error since the solution is not unique, given the probe vehicle data.

Starting with the new interval $t\in[10,18]$, we notice the artifacts in the observer estimation in case of the model mismatch, and the resulting error increases. For the PINN model, the error stagnates until the learned free flow velocity converges to the new value, and afterwards, the error begins to decrease rapidly. A contributing factor to the reduced error can also be explained by the increase in density of the lower boundary conditions. In this time frame, in the case of a large model mismatch, we can see that PINN achieves a strictly lower error by adapting to the changing free-flow velocity.

In the last interval $t\in[18,30]$, the error starts to increase for the PINN model since the learned free flow velocity v_f has an initial high discrepancy. In contrast, the error decreases for the observer due to the reduced model discrepancy. From $t\in[21,30]$, both the observer and the PINN model errors decrease gradually with time, achieving comparable performance. At $t=21$min, the PINN model no longer has any velocity or density measurements from the previous free flow velocity in the data window, which explains why the error starts to decrease. A lower δ_d or δ_v would have contributed to a faster recovery in error after the free flow shift at the price of a lower robustness. From the following simulations, we can conclude that in the case of perfect model knowledge, the observer has superior performance; however, in the case of high model error, the

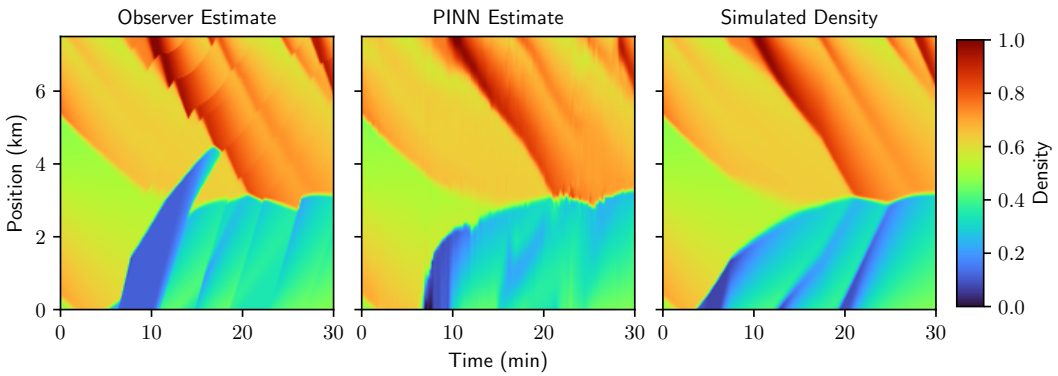

Fig. 5. Reconstructed and simulated data, in the case of varying free flow velocity, leading to areas of imperfect model knowledge.

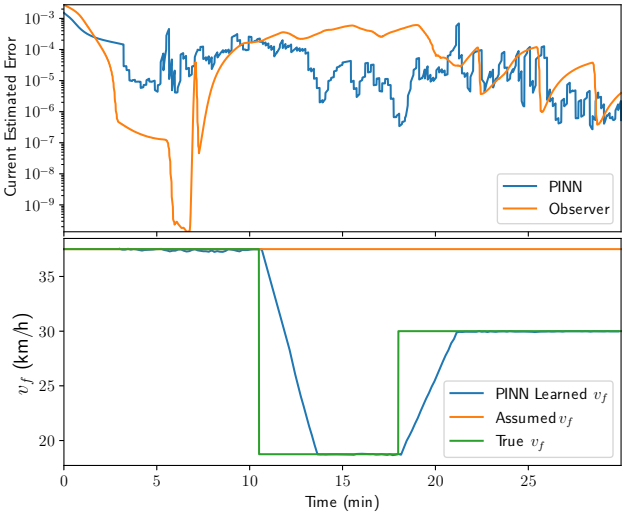

Fig. 6. CEE for the observer and PINN for a varying v_f.

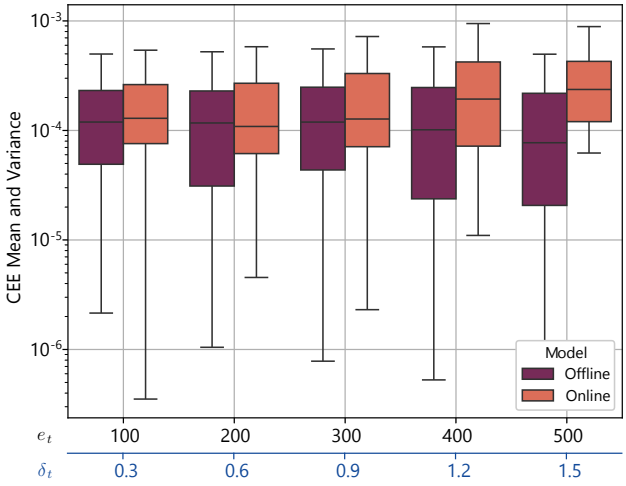

Fig. 7. CEE for different training iterations e_t, one model trained in the 'Online' case, such that δ_t increases with e_t, and another model trained in the 'Offline' case, such that $\delta_t = 0.3$ independent of e_t.

PINN model outperforms the observer, and with a low model error, the performances are similar.

2) The effect of parameter changes on performance: Fig. 7 presents the comparison of results after training two models, an 'Online' model, in which δ_t depends on the number of training iterations, and an 'Offline' model, where $\delta_t = 0.3$ is kept constant. Here we can see that the mean CEE error for both 'Online' and 'Offline' models is similar, when $e_t \in [100, 300]$. However, when $e_t \geq 300$, the 'Online' model error starts to increase as a consequence of the delayed estimation caused by a longer training time δ_t, and the 'Offline' model error decreases due to improved convergence. Hence, Fig. 7 highlights the trade-off appearing in the online case, where the increased number of iterations leads to increased error, contrary to the traditional scenario, when the error is expected to decrease.

B. SUMO simulation

SUMO simulation allows testing our model for robustness in a more realistic scenario. The exact dynamics of the macroscopic behavior of the system are unknown. As such the velocity function $v(\rho)$ has to be learned, adhering to the constraints $v(0) = v_f$, $v(1) = 0$ and $\partial_\rho v < 0$.

The PINN estimate of the current density, the actual density produced by a SUMO simulation, and the CEE of the PINN can be seen in Fig. 8. The PINN reaches initial convergence at around $t = 0.5min$ after which the CEE oscillates around the point $CEE \approx 0.01$. A possible cause for the oscillations is the discontinuities in the data. For example, area 'A' in Fig. 8 marks the increase of the error followed by its decrease depicted by the area 'B'. Area 'A' does not have any probe vehicle data and, as time progresses, we can see how an erroneous higher density green 'tail', is formed, due to previous errors being propagated. As the tail is forming, we can see the error increase with time until approximately $t \approx 4.4min$, where a probe vehicle arrives to supply new measurements. Subsequently, the tail disappears, and the error rapidly declines.

Another notable behavior is the short spike in error appearing at time $t \approx 3.6$min and $t \approx 4.7$min. Fig. 8 shows the error at $t \approx 3.6$min caused by the road segment $x \in [0, 1.0]$ having a misestimated density compared to its neighborhood. This potentially could be caused by either a lack of data

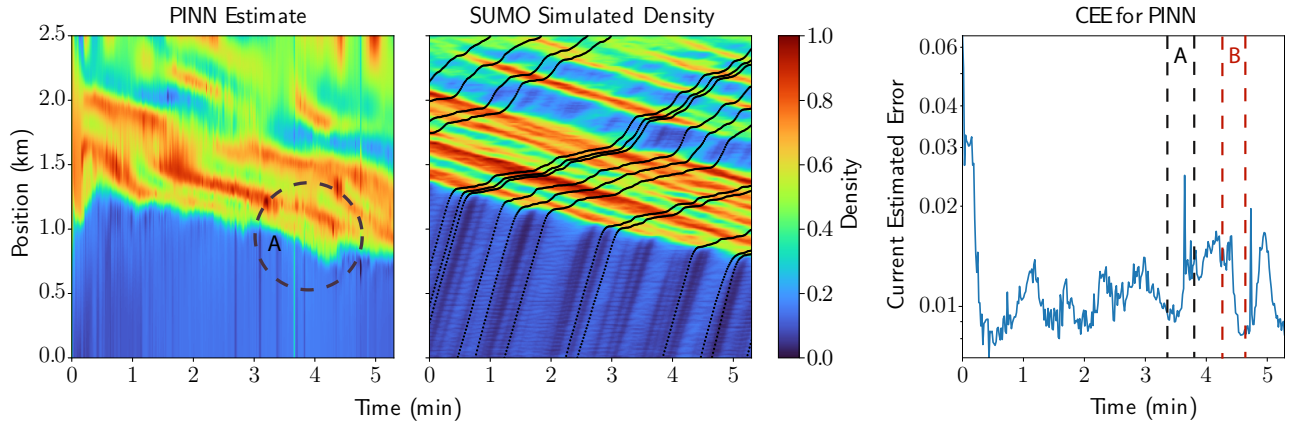

Fig. 8. The estimation of the current time density is shown on the left. The corresponding SUMO simulation with dots marking the measurements from the probe vehicles is depicted in the central figure. On the right, we see the corresponding CEE between the PINN estimates and the SUMO simulation.

from probing vehicles or insufficient training iterations not allowing the information to propagate across the area, or a combination of the two.

V. Conclusion

The demand for accurate real-time traffic density estimations is rising due to the need for effective autonomous transport and the increased reliance on the implementation of traffic control strategies. To address the issues arising from moving to real-time traffic control, we propose a framework for online traffic density estimation using PINNs. The conducted numerical experiments highlight the robustness of the PINN in the case of a time-varying traffic model, when the PINN outperforms a classical open-loop observer. They also show the negative effects of training time on the estimation quality, a property only apparent in the online scenario. The application to data generated using SUMO shows that the proposed method performs well, even in the case of an unknown high-fidelity dynamical model.

Future research on the topic should focus on improving the speed of convergence of the PINN to reduce the impact of the training time, investigating the introduction of uncertainty quantification to identify areas of high and low reliability of the estimate, and improving the adaptability of the network by considering the use of dynamic weights on the probe vehicle measurements.

References

[1] H. Ritchie, "Cars, planes, trains: where do CO2 emissions from transport come from?," *Our World in Data*, 2020. https://ourworldindata.org/co2-emissions-from-transport.

[2] M. Čičić and K. H. Johansson, "Traffic regulation via individually controlled automated vehicles: a cell transmission model approach," in *International Conference on Intelligent Transportation Systems*, 2018.

[3] M. L. Delle Monache, T. Liard, and al., "Feedback control algorithms for the dissipation of traffic waves with autonomous vehicles," *Computational Intelligence and Optimization Methods for Control Engineering*, 2019.

[4] A. Ferrara, S. Sacone, S. Siri, *et al.*, *Freeway traffic modelling and control*, vol. 585. Springer, 2018.

[5] P. I. Richards, "Shock waves on the highway," *Operations Research*, 1956.

[6] M. J. Lighthill and G. B. Whitham, "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*, 1955.

[7] P. A. Lopez *et al.*, "Microscopic traffic simulation using sumo," in *2018 21st international conference on intelligent transportation systems (ITSC)*, 2018.

[8] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Reviews Physics*, 2021.

[9] X. Di, R. Shi, Z. Mo, and Y. Fu, "Physics-informed deep learning for traffic state estimation: A survey and the outlook," *Algorithms*, 2023.

[10] M. Barreau, M. Aguiar, J. Liu, and K. H. Johansson, "Physics-informed learning for identification and state reconstruction of traffic density," in *60th IEEE Conference on Decision and Control*, 2021.

[11] A. J. Huang and S. Agarwal, "Physics informed deep learning: Applications in transportation," *arXiv preprint arXiv:2302.12336*, 2023.

[12] Y. Wang and M. Papageorgiou, "Real-time freeway traffic state estimation based on extended kalman filter: a general approach," *Transportation Research Part B: Methodological*, 2005.

[13] M. Barreau, A. Selivanov, and K. H. Johansson, "Dynamic traffic reconstruction using probe vehicles," in *59th IEEE Conference on Decision and Control*, 2020.

[14] C. M. Dafermos, *Hyperbolic Conservation Laws in Continuum Physics; 3rd ed.* Springer, 2010.

[15] B. D. Greenshields, J. R. Bibbins, W. Channing, and H. H. Miller, "A study of traffic capacity," in *Highway research board proceedings*, Washington, DC, 1935.

[16] M. Barreau, J. Liu, and K. H. Johansson, "Learning-based state reconstruction for a scalar hyperbolic PDE under noisy lagrangian sensing," in *Learning for Dynamics and Control*, PMLR, 2021.

[17] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, 1991.

[18] M. Barreau and H. Shen, "Accuracy and robustness of weight-balancing methods for training PINNs," *arXiv preprint arXiv:2501.18582*, 2025.

[19] M. L. Delle Monache, T. Liard, B. Piccoli, R. Stern, and D. Work, "Traffic reconstruction using autonomous vehicles," *SIAM Journal on Applied Mathematics*, 2019.

[20] R. J. LeVeque, *Numerical methods for conservation laws (2. ed.)*. Lectures in mathematics, Birkhäuser, 1992.