Sketching the Dynamics of Traffic: Data-Efficient Predictive Connected Cruise Control

Minghao Shen, *Graduate Student Member, IEEE*, Jeff Scruggs, *Senior Member, IEEE*, and Gábor Orosz, *Senior Member, IEEE*

***Abstract*—Vehicle-to-everything (V2X) communication enables connected cruise control by providing upstream traffic information beyond the line of sight. However, practical V2X deployments face challenges including packet loss, limited training data, and domain shifts between different driving conditions. In this paper, we propose a data-driven prediction framework that addresses these challenges through memory-efficient online adaptation. The ego vehicle predicts the speed of the vehicle immediately ahead based on the speeds of connected vehicles in the upstream traffic. We derive an input-output model from state-space car-following dynamics and develop a Gaussian sketching algorithm that enables efficient online adaptation, reducing per-step update complexity and memory compared to the full-dimension recursive least squares (RLS) algorithm. We validate the framework using real-world traffic data under challenging scenarios: cross-domain transfer, limited warmstart data, and V2X packet loss. The proposed Gaussian sketch method achieves a 34.4% improvement under input distribution shift, outperforming both non-adaptive and full RLS baselines. It also reaches up to 92.1% improvement with limited warmstart data. Crucially, the method maintains a 30.7% improvement even under 50% packet loss, demonstrating robustness to realistic V2X communication impairments.**

***Index Terms*—Data-driven control, predictive control, vehicle-to-everything communication, dimension reduction.**

I. Introduction

ADAPTIVE cruise control, a technology dating back to the 1990s, automates longitudinal motions of vehicles, while responding to the motion of the vehicle immediately ahead. Vehicle-to-everything (V2X) connectivity enables vehicles to communicate beyond the line of sight, supporting cooperative maneuvers and more informed decision making in traffic [1], [2], [3]. This can provide significant benefits in terms of traffic throughput and energy efficiency [4], [5], [6], [7], [8], [9].

Nevertheless, the effectiveness of such technologies depends on the penetration rates of connected vehicles. In addition, cooperative controls require the management of participating vehicles. These requirements pose significant challenges for the widespread adoption of connectivity. Consequently, an important question arises: Can we benefit from V2X communication even when both the penetration level of connectivity and the degree of cooperation are low? Addressing this question could lead to robust and scalable V2X-based solutions for intelligent transportation systems and stimulate further research and investment in connectivity and automation.

With sparse penetration and low cooperation levels, the crucial role of connectivity is to provide additional information about upstream traffic conditions beyond the line of sight [1]. Connected cruise control (CCC) is designed with this concept in mind [10]. It does not require all vehicles to be connected, nor does it necessitate controlling other vehicles. There are two main design approaches for CCC: reactive and predictive [11]. Reactive CCC utilizes explicit feedback control laws that respond to current traffic conditions. In contrast, predictive CCC anticipates future traffic conditions and optimizes control actions based on these predictions. While the explicit feedback control law in reactive CCC facilitates analysis of stability and performance, the expressiveness of the controller parameterization may also limit performance. Predictive CCC offers greater flexibility in control actions, while the major challenge lies in accurately predicting future traffic conditions, specifically the future motion of nearby vehicles while leveraging additional information from connectivity.

To accurately predict the future motions of nearby vehicles, a model is required that captures the relationship between the states of upstream traffic vehicles. Traditional microscopic traffic models such as the optimal velocity model (OVM) [12], [13] the intelligent driver model (IDM) [11], [14], or Newell's model [15], [16] could be employed. However, under conditions of sparse connectivity and dynamically varying traffic, the information available for model identification is often limited. Instead, expanding our previous work [17], we propose a data-driven approach that directly learns the relationship between the input (the speeds of vehicles in upstream traffic) and the output (the speed of the vehicle immediately in front of the ego vehicle). We develop an input-output model that circumvents the challenges of identifying the number and states of hidden vehicles, focusing instead on the head-to-tail dynamics between the input and output [18]. Based on the behavioral theory of linear systems, this data-driven approach can capture essential dynamics without explicit model parameterization, given sufficient input-output data [19], [20], [21], [22].

Received 9 April 2025; revised 26 February 2026; accepted 21 April 2026. Date of publication 27 May 2026; date of current version 4 September 2026. This work was supported by the University of Michigan's Center for Connected and Automated Transportation through U.S. Department of Transportation (DOT) under Grant 69A3551747105. The Associate Editor for this article was Y. He. *(Corresponding author: Minghao Shen.)*

Minghao Shen is with the Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: mhshen@umich.edu).

Jeff Scruggs is with the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: jscruggs@umich.edu).

Gábor Orosz is with the Department of Mechanical Engineering and the Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: orosz@umich.edu).

Digital Object Identifier 10.1109/TITS.2026.3694599

1558-0016 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

The data-driven approach necessitates efficient storage and processing of large-scale online streaming data. According to Willems' fundamental lemma, a cornerstone in behavioral theory, the input data must be persistently exciting to accurately capture the dynamics. Additionally, to address time-varying traffic conditions, data must be continuously collected and adapted to changes, significantly increasing the burden on real-time computation for predictions. A critical challenge is balancing the richness of the data while minimizing its size. One option is to implement singular value decomposition (SVD), performing dimension reduction by discarding data corresponding to small singular values [23]. However, this approach demands iterative, computationally intensive SVD calculations as new data arrives. In this paper, we propose a novel memory sketching method to optimize data storage and processing, effectively balancing prediction accuracy with computational efficiency. Sketching, a technique that compresses data into a smaller size while preserving essential information, is widely used in machine learning and data mining, and has proven effective in reducing the computational costs of large-scale optimization problems [24], [25], [26]. Here, we develop sketching for the data-driven predictive control of traffic systems with sparse V2X communication and demonstrate the effectiveness of our proposed method through numerical simulations.

Our method offers a scalable, efficient, and accurate solution for traffic prediction tailored to the needs of individual connected automated vehicles (CAVs). Our contributions are listed as follows:

1) Theoretical: We derive an input-output model for traffic prediction that captures head-to-tail dynamics without requiring knowledge of the number or states of intermediate vehicles.
2) Algorithmic: We propose a memory sketching algorithm for online adaptation of the prediction model. The key idea is to update the model in a compressed representation, so each step depends on the sketch size rather than the full regressor size, while also avoiding repeated full-history refitting.
3) Experimental: We validate the proposed framework using real-world traffic data and demonstrate its robustness to domain shifts and limited training data.

The rest of the paper is structured as follows. In Sec. II we lay down the key principles of sparse V2X-based traffic prediction. In Sec. III we establish the link between state-space traffic models and input-output traffic models. These "ground-truth" models are used to generate synthetic data for the data-driven traffic predictions developed in Sec. IV and also to evaluate the performance of the data-based methods. Memory sketching is presented in Sec. V and the results of different prediction methods are compared via numerical simulations in Sec. VI. The results are concluded and future research directions are highlighted in Sec. VII.

II. Prediction With Sparse V2X Communication

Vehicle-to-everything communication enables connected vehicles to exchange information, such as speed, acceleration, and position, with each other. In this paper, we consider the problem of speed prediction with sparse V2X communication. As is shown in Fig. 1, the ego CAV is following a chain of human-driven vehicles. Some of those are equipped with V2X communication, so the ego vehicle has access to the speed of these connected human-driven vehicles (CHVs), such as the vehicle L in Fig. 1. In motion planning and control, the ego CAV needs to predict the speed of the human-driven vehicle (HV) immediately in the front (the red vehicle 1 in the figure). In dense traffic, the speeds of vehicles are highly correlated, and the speed of the vehicle in the front is a key factor to determine the speed of the ego vehicle. Therefore, the speed information of vehicles in the upstream traffic is crucial to accurately predict the speed of the vehicle immediately ahead.

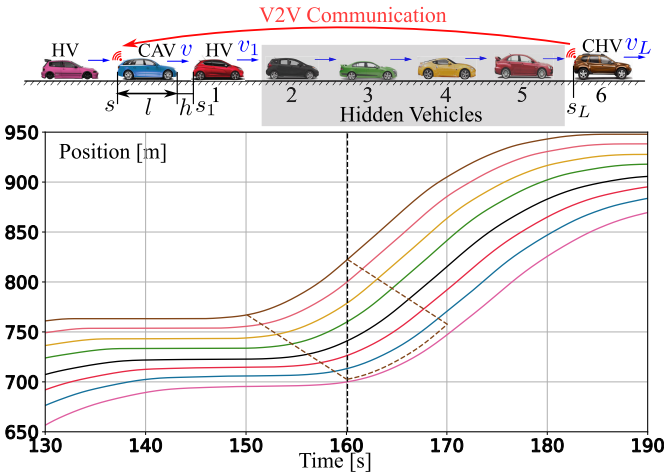


Fig. 1. The concept of V2X connectivity-based traffic prediction.

Speed prediction of downstream vehicles based on the speed of upstream vehicles has been widely studied in the literature. Existing methods can be broadly categorized into model-based and data-driven approaches. Classical traffic models like Newell's model [15], [27], [28], optimal velocity model (OVM) [12], [13], [29], and intelligent driver model [11], [14] are widely used to model the car-following behaviors in traffic. However, these models require the knowledge of the number of vehicles between the CAV and the CHV as well as the corresponding driver parameters. Model-based methods like Kalman filtering on Newell's model or OVM require explicit parameter identification and often assume fixed traffic structures [28]. Such information is typically not available in practice, and it can vary over time as vehicles merge into and exit the current lane. As illustrated in Fig. 1, one may utilize that there is an (approximately constant) wave propagation speed, and then the position trajectories become (approximate) copies of the trajectory of the head vehicle shifted along the wave characteristics. The accuracy of such predictions is, however, limited and the continuity of the prediction is not guaranteed [28].

On the other hand, data-driven trajectory reconstruction methods aim to estimate the full state of all vehicles from sparse observations [16]. While effective for traffic monitoring, these methods are computationally intensive and focus on reconstruction rather than real-time prediction for control.

Authorized licensed use limited to: University of Michigan Library. Downloaded on November 03,2026 at 19:37:08 UTC from IEEE Xplore. Restrictions apply.

TABLE I
TABLE OF NOTATIONS

Symbol	Description
L	Number of vehicles in the upstream traffic
h_i, v_i	Headway between vehicles i and $i+1$, speed of vehicle i
$T^{\mathrm{p}}, T^{\mathrm{f}}$	Past and future horizon lengths
u, y	Input (head vehicle speed) and output (preceding vehicle speed)
$u^{\mathrm{p}}, y^{\mathrm{p}}$	Past input and output trajectories
$u^{\mathrm{f}}, y^{\mathrm{f}}$	Future input and output trajectories
M	Regressor dimension (row dimension of Ξ)
N	Number of trajectory columns / historical data points
S	Sketch size (dimension of the projection)
Π	Projection matrix for memory sketching
ξ	Data vector $\xi = [u^{\mathrm{p}\top}, y^{\mathrm{p}\top}, \hat{u}^{\mathrm{f}\top}]^\top$
Φ, Ψ	Input-output and autoregressive solution operators

Instead of assuming the knowledge of a detailed car-following model for each vehicle ahead or reconstructing full states, we propose a data-driven approach for prediction. We utilize the information from V2X connectivity and identify the traffic dynamics online. We demonstrate that such data-driven approach can capture the essential wave propagation dynamics of the traffic system and make accurate prediction of the speed of the vehicle immediately ahead. This allows us to develop an online adaptive framework that is robust to model mismatches and changing traffic conditions.

In this paper, we address the problems of speed prediction with sparse V2X communication. We first build a dynamical model for the upstream traffic assuming the full knowledge of the states and parameters of the vehicles and refer to this as the "ground truth" traffic model. Then we propose data-driven methods to address the lack of information due to the sparsity of connectivity. We show that the data-driven model can capture the essential dynamics of the ground truth system.

III. Dynamical System for Speed Prediction

In this section, we build the ground truth traffic model assuming full knowledge of the states and parameters of the vehicles in the upstream traffic. We start from a state-space model which is directly derived from classical car-following models, and then build an input-output model that is utilized to evaluate the data-driven prediction model in later sections. The key variables used in the paper are listed in Table I.

A. State-Space Traffic Model

Assume that there are L vehicles in the upstream traffic, and we have full knowledge of their states and parameters. Then, a state-space model describing the motion of the vehicles between vehicle L and vehicle 1 can be formulated as

$$\begin{aligned}\dot{h}_i(t) &= -v_i(t) + v_{i+1}(t),\\ \dot{v}_i(t) &= f_i\big(h_i(t-\zeta_i), v_i(t-\zeta_i), v_{i+1}(t-\zeta_i)\big),\end{aligned} \tag{1}$$

for $i = 1, \dots, L-1$, where h_i is the headway between vehicle i and vehicle $i+1$ and v_i is the speed of vehicle i. The function f_i and the time delay ζ_i are the car-following policy and the reaction time of the driver of vehicle i, respectively. One example is the optimal velocity model (OVM) [12]:

$$f_i\big(h_i, v_i, v_{i+1}\big) = \alpha_i\big(V_i(h_i) - v_i\big) + \beta_i\big(W(v_{i+1}) - v_i\big). \tag{2}$$

The range policy

$$V_i(h_i) = \begin{cases} 0, & \text{if} \quad h_i < h_{\mathrm{st},i} \\ \kappa_i(h_i - h_{\mathrm{st},i}), & \text{if} \quad h_{\mathrm{st},i} \le h_i \le h_{\mathrm{st},i} + v_{\max}/\kappa_i \\ v_{\max}, & \text{if} \quad h_i > h_{\mathrm{st},i} + v_{\max}/\kappa_i \end{cases} \tag{3}$$

determines the desired speed given the headway h_i. The desired speed is 0 when the headway is less than the standstill headway $h_{\mathrm{st},i}$ and then it grows linearly with gradient κ_i until it reaches the speed limit $v_{\max}$. The speed policy $W(v_{i+1}) = \min\{v_{i+1}, v_{\max}\}$ is introduced to avoid following a speeding vehicle.

In the context of data-driven prediction, all the data are collected in discrete time. So we discretize the continuous-time model (1) using Euler's method, yielding

$$\begin{aligned} h_i[k+1] &= h_i[k] - \Delta t v_i[k] - \frac{1}{2}\Delta t^2 a_i[k] + \Delta t v_{i+1}[k],\\ v_i[k+1] &= v_i[k] + \Delta t a_i[k],\\ a_i[k] &= f_i\big(h_i[k-r_i], v_i[k-r_i], v_{i+1}[k-r_i]\big). \end{aligned} \tag{4}$$

One may show that system (4) possesses the equilibrium

$$h_i[k] \equiv h_i^*, \quad v_i[k] \equiv v^*, \quad a_i[k] \equiv 0, \tag{5}$$

for $i = 1, \dots, L-1$. For the OVM (2) this yields $V(h_i^*) = v^*$ when $h_{\mathrm{st}} \le h_i^* \le h_{\mathrm{st}} + v_{\max}/\kappa_i$, cf. (3). We define the deviation variables

$$\tilde{h}_i = h_i - h_i^*, \quad \tilde{v}_i = v_i - v_i^*, \quad \tilde{a}_i = a_i, \tag{6}$$

and linearize the system (4) around the equilibrium, yielding

$$\begin{aligned} \tilde{h}_i[k+1] &= \tilde{h}_i[k] - \Delta t \tilde{v}_i[k] - \frac{1}{2}\Delta t^2 \tilde{a}_i[k] + \Delta t \tilde{v}_{i+1}[k],\\ \tilde{v}_i[k+1] &= \tilde{v}_i[k] + \Delta t \tilde{a}_i[k],\\ \tilde{a}_i[k] &= \partial_{h_i} f_i\left(h_i^*, v^*, v^*\right)\tilde{h}_i[k-r_i]\\ &\quad + \partial_{v_i} f_i\left(h_i^*, v^*, v^*\right)\tilde{v}_i[k-r_i]\\ &\quad + \partial_{v_{i+1}} f_i\left(h_i^*, v^*, v^*\right)\tilde{v}_{i+1}[k-r_i]. \end{aligned} \tag{7}$$

For the OVM (2) we have

$$\begin{aligned} \partial_{h_i} f_i\left(h_i^*, v^*, v^*\right) &= \alpha_i\kappa_i,\\ \partial_{v_i} f_i\left(h_i^*, v^*, v^*\right) &= -(\alpha_i + \beta_i),\\ \partial_{v_{i+1}} f_i\left(h_i^*, v^*, v^*\right) &= \beta_i. \end{aligned} \tag{8}$$

In fact, the constants α_i, β_i and κ_i can be used to parametrize the linearization of any car-following model of the form (1) [30]. We refer to this as the OVM parametrization and use it in the rest of the paper.

System (7) is a discrete-time linear delayed system. By defining the state $x_i \in \mathbb{R}^{3+r_i}$ as

$$x_i[k] = \begin{bmatrix} \tilde{h}_i[k] \\ \tilde{v}_i[k] \\ \tilde{a}_i[k] \\ \tilde{a}_i[k-1] \\ \vdots \\ \tilde{a}_i[k-r_i] \end{bmatrix}, \tag{9}$$

we can build state-space model

$$x_i[k+1] = A_i x_i[k] + B_i \tilde{v}_{i+1}[k],$$

$$y_i[k] = C_i x_i[k] + D_i \tilde{v}_{i+1}[k], \tag{10}$$

where

$$A_i = \begin{bmatrix} 1 & -\Delta t & -\frac{\Delta t^2}{2} & 0 & \cdots & 0 & 0 \\ 0 & 1 & \Delta t & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & & & \\ \vdots & \vdots & \vdots & & \ddots & & \\ 0 & 0 & 0 & & & 1 & 0 \\ \alpha_i \kappa_i & -\alpha_i - \beta_i & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}, \tag{11}$$

$$B_i = \begin{bmatrix} \Delta t & 0 & \cdots & 0 & \beta_i \end{bmatrix}^\top,$$

$$C_i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \end{bmatrix}, \quad D_i = 0. \tag{12}$$

Then we define $x = \begin{bmatrix} x_1^\top & x_2^\top & \cdots & x_{L-1}^\top \end{bmatrix}^\top$, $u = \tilde{v}_L$, and $y = y_1$ where $x \in \mathbb{R}^{n_x}$ and $n_x = 3(L-1) + \sum_{i=1}^{L-1} r_i$. Therefore,

$$\begin{aligned} x[k+1] &= Ax[k] + Bu[k], \\ y[k] &= Cx[k] + Du[k], \end{aligned} \tag{13}$$

where

$$A = \begin{bmatrix} A_1 & B_1 C_1 & & & \\ & A_2 & B_2 C_2 & & \\ & & \ddots & \ddots & \\ & & & & A_{L-1} \end{bmatrix}, \tag{14}$$

$$B = \begin{bmatrix} 0 & \cdots & 0 & B_{L-1}^\top \end{bmatrix}^\top, \tag{15}$$

$$C = \begin{bmatrix} C_1 & 0 & \cdots & 0 \end{bmatrix}, \quad D = 0. \tag{16}$$

This state-space model connects the speed of the vehicle L and the speed of the vehicle 1 in the upstream traffic. We refer this to *head-to-tail dynamics* [31]. This linear time-invariant (LTI) model can be used for predictive control. However, in mixed traffic consisting of both connected and non-connected vehicles, the states and the parameters of the vehicles upstream are not necessarily available. In fact, often even the number of vehicles L is not available, which poses great challenges for model identification and state estimation.

B. Input-Output Traffic Model

Through V2X connectivity, the ego vehicle can access the speed of the head vehicle. An input-output model bypasses the difficulty of identifying the number of hidden vehicles, the model states and parameters, and focuses on linking the input (the speeds of the head vehicle), and the output (the speed of the vehicle immediately in front of the ego vehicle).

We use snippets of input-output data to characterize the system dynamics using the input data and output data within the past horizon T^{p}:

$$u^{\mathrm{p}}[k] = \begin{bmatrix} u[k - T^{\mathrm{p}} + 1], u[k - T^{\mathrm{p}} + 2], \ldots, u[k] \end{bmatrix}^\top, \tag{17}$$

$$y^{\mathrm{p}}[k] = \begin{bmatrix} y[k - T^{\mathrm{p}} + 1], y[k - T^{\mathrm{p}} + 2], \ldots, y[k] \end{bmatrix}^\top. \tag{18}$$

Once we know the future input

$$u^{\mathrm{f}}[k] = \begin{bmatrix} u[k+1], u[k+2], \ldots, u[k + T^{\mathrm{f}}] \end{bmatrix}^\top, \tag{19}$$

we can predict the future output

$$y^{\mathrm{f}}[k] = \begin{bmatrix} y[k+1], y[k+2], \ldots, y[k + T^{\mathrm{f}}] \end{bmatrix}^\top. \tag{20}$$

Namely, for the LTI system (13), we can construct a multi-step predictor:

$$\begin{aligned} y^{\mathrm{p}}[k] &= \mathcal{O}_{T^{\mathrm{p}}} x[k - T^{\mathrm{p}} + 1] \\ &\quad + \mathcal{T}(D, CB, CAB, \ldots, CA^{T^{\mathrm{p}}-2}B) u^{\mathrm{p}}[k], \end{aligned} \tag{21}$$

$$\begin{aligned} y^{\mathrm{f}}[k] &= \mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}} x[k - T^{\mathrm{p}} + 1] \\ &\quad + \mathcal{O}_{T^{\mathrm{f}}} \mathcal{K}_{T^{\mathrm{p}}}(A)(I_{T^{\mathrm{p}}} \otimes B) u^{\mathrm{p}}[k] \\ &\quad + \mathcal{T}(D, CB, CAB, \ldots, CA^{T^{\mathrm{f}}-2}B) u^{\mathrm{f}}[k], \end{aligned} \tag{22}$$

where the operators $\mathcal{O}_T$, $\mathcal{T}$ and $\mathcal{K}_T$ are defined as

$$\mathcal{O}_T = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T-1} \end{bmatrix}, \tag{23a}$$

$$\mathcal{T}(A_1, A_2, \ldots, A_n) = \begin{bmatrix} A_1 & 0 & \cdots & 0 \\ A_2 & A_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_n & A_{n-1} & \cdots & A_1 \end{bmatrix}, \tag{23b}$$

$$\mathcal{K}_T(A) = \begin{bmatrix} A^{T-1}, A^{T-2}, \ldots, A, I \end{bmatrix}. \tag{23c}$$

Here $\mathcal{O}_T$ operator generalizes the observability matrix. When the system is observable and $T \geq n_x$, where n_x is the dimension of the state in (13), $\mathcal{O}_T(A)$ has full column rank. We keep this assumption for the rest of the paper, since we consider a minimum phase system for the input-output model. The operator $\mathcal{T}$ creates a block Toeplitz matrix from the input matrices, and operator $\mathcal{K}_T$ creates a Krylov matrix from the matrix A. Note that $\otimes$ denotes the Kronecker product.

In (21), the initial state $x[k - T^{\mathrm{p}} + 1]$ is unknown. However, we can prove the following lemma, which shows the future output $y^{\mathrm{p}}[k]$ is independent of the initial state $x[k - T^{\mathrm{p}} + 1]$ under certain condition.

Theorem 1: In (21) and (22), with given $u^{\mathrm{p}}[k]$, $u^{\mathrm{f}}[k]$ and $y^{\mathrm{p}}[k]$, the future output $y^{\mathrm{f}}[k]$ is independent of the initial state $x[k - T^{\mathrm{p}} + 1]$ if and only if there exists $G \in \mathbb{R}^{T^{\mathrm{f}} n_x \times n_x}$ such that

$$\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}} = G \mathcal{O}_{T^{\mathrm{p}}}. \tag{24}$$

Proof: The solution set of $x[k - T^{\mathrm{p}} + 1]$ is

$$X = \{x_0 + z | z \in \mathrm{null}(\mathcal{O}_{T^{\mathrm{p}}}), x_0 \text{ is solution to (21)}\}. \tag{25}$$

The independence of y^{f} on $x[k - T^{\mathrm{p}} + 1]$ is equivalent to

$$\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}} z = 0, \ \forall z \in \mathrm{null}(\mathcal{O}_{T^{\mathrm{p}}}), \tag{26}$$

which means the null space of $\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}}$ is contained in the null space of $\mathcal{O}_{T^{\mathrm{p}}}$:

$$\mathrm{null}(\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}}) \supseteq \mathrm{null}(\mathcal{O}_{T^{\mathrm{p}}}). \tag{27}$$

Since the row space is the orthogonal complement of the null space, the row space of $\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}}$ is contained in the row space of $\mathcal{O}_{T^{\mathrm{p}}}$:

$$\mathrm{row}(\mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}}) \subseteq \mathrm{row}(\mathcal{O}_{T^{\mathrm{p}}}), \tag{28}$$

which gives (24). □

Extracting $x[k - T^{\mathrm{p}} + 1]$ from (21) gives

$$x[k - T^{\mathrm{p}} + 1] = \mathcal{O}_{T^{\mathrm{p}}}^{\dagger} \big(y^{\mathrm{p}}[k]$$

$$- \mathcal{T}(D, CB, CAB, \ldots, CA^{T^{\mathrm{p}}-2}B)u^{\mathrm{p}}[k]\big), \quad (29)$$

where † denotes the Moore-Penrose inverse (pseudoinverse). Substituting this into (22), we have the multi-step predictor

$$\begin{aligned} y^{\mathrm{f}}[k] = \big(&- \mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}} \mathcal{O}^{\dagger}_{T^{\mathrm{p}}} \mathcal{T}(D, CB, CAB, \ldots, CA^{T^{\mathrm{p}}-2}B) \\ &+ \mathcal{O}_{T^{\mathrm{f}}} \mathcal{K}_{T^{\mathrm{p}}}(A)(I_{T^{\mathrm{p}}} \otimes B)\big) u^{\mathrm{p}}[k] \\ &+ \mathcal{O}_{T^{\mathrm{f}}} A^{T^{\mathrm{p}}} \mathcal{O}^{\dagger}_{T^{\mathrm{p}}} y^{\mathrm{p}}[k] \\ &+ \mathcal{T}(D, CB, CAB, \ldots, CA^{T^{\mathrm{f}}-2}B) u^{\mathrm{f}}[k]. \end{aligned} \quad (30)$$

More concisely we can write

$$y^{\mathrm{f}}[k] = \Phi \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \\ u^{\mathrm{f}}[k] \end{bmatrix} = \Phi \xi[k]. \quad (31)$$

The solution operator Φ gives the linear relationship between the input $\xi[k]$ and the output $y^{\mathrm{f}}[k]$. Now our goal becomes to find such relationship from data.

IV. Data-Driven Prediction

The previous section built an input-output traffic model assuming that the car-following dynamics of the upstream vehicles are known. However, in real-world scenarios such knowledge is typically not available. Instead, we can determine the head-to-tail dynamics directly from data. We refer to this as *data-driven prediction*.

A. Preliminaries on Data-Driven Prediction

Consider a dataset of input-output trajectory snippets:

$$\mathcal{D} = \bigcup_{j=1}^{J} \left\{ u^j[1 : L_j], y^j[1 : L_j] \right\}, \quad (32)$$

where the j-th trajectory snippet has length L_j, and we introduced the notation $u^j[1 : L_j] = \left[u^j[1]^\top, \ldots, u^j[L_j]^\top\right]^\top$. These snippets are not necessarily disjoint nor overlapped. For each snippet, we extract the input data $u^j[1 : L_j]$ into

$$\begin{aligned} U_j &= \mathcal{H}_T\left(u^j[1 : L_j]\right) \\ &= \begin{bmatrix} u^j[1] & u^j[2] & \cdots & u^j[L_j - T + 1] \\ u^j[2] & u^j[3] & \cdots & u^j[L_j - T + 2] \\ \vdots & \vdots & & \vdots \\ u^j[T^{\mathrm{p}}] & u^j[T^{\mathrm{p}}+1] & \cdots & u^j[L_j - T^{\mathrm{f}}] \\ u^j[T^{\mathrm{p}}+1] & u^j[T^{\mathrm{p}}+2] & \cdots & u^j[L_j - T^{\mathrm{f}}+1] \\ \vdots & \vdots & & \vdots \\ u^j[T] & u^j[T+1] & \cdots & u^j[L_j] \end{bmatrix} \\ &= \begin{bmatrix} U^{\mathrm{p}}_j \\ U^{\mathrm{f}}_j \end{bmatrix}, \end{aligned} \quad (33)$$

where $T = T^{\mathrm{p}} + T^{\mathrm{f}}$, U^{p}_j is the first T^{p} rows of U_j, summarizing all the data corresponding to u^{p} within the snippet. Similarly U^{f}_j is the last T^{f} rows of U_j, corresponding to u^{f}. Similarly, we extract the output data $y^j[1 : L_j]$ into

$$Y_j = \mathcal{H}_T\left(y^j[1 : L_j]\right) = \begin{bmatrix} Y^{\mathrm{p}}_j \\ Y^{\mathrm{f}}_j \end{bmatrix}, \quad (34)$$

where Y^{p}_j is the first T^{p} rows of Y_j, corresponding to y^{p}, and Y^{f}_j is the last T^{f} rows of Y_j, corresponding to y^{f}.

Combining all snippets in the dataset, we organize the training data into the following matrices:

$$\begin{aligned} U^{\mathrm{p}} &= \begin{bmatrix} U^{\mathrm{p}}_1 & U^{\mathrm{p}}_2 & \cdots & U^{\mathrm{p}}_J \end{bmatrix}, \\ U^{\mathrm{f}} &= \begin{bmatrix} U^{\mathrm{f}}_1 & U^{\mathrm{f}}_2 & \cdots & U^{\mathrm{f}}_J \end{bmatrix}, \\ Y^{\mathrm{p}} &= \begin{bmatrix} Y^{\mathrm{p}}_1 & Y^{\mathrm{p}}_2 & \cdots & Y^{\mathrm{p}}_J \end{bmatrix}, \\ Y^{\mathrm{f}} &= \begin{bmatrix} Y^{\mathrm{f}}_1 & Y^{\mathrm{f}}_2 & \cdots & Y^{\mathrm{f}}_J \end{bmatrix}. \end{aligned} \quad (35)$$

The idea of data-driven prediction is that, when the input signal persistently excites the system, the input-output pair can fully characterize the system dynamics. The following definition is put forward to describe the richness of the input signal.

Definition 1: Consider the input sequences $u^j[1 : L_j]$ for $j = 1, \ldots, J$. Let $K \le L_j$ be a positive integer for all j. The input sequences are *collectively persistently exciting* of order K if the matrix

$$\begin{bmatrix} \mathcal{H}_K(u^1[1 : L_1]) & \cdots & \mathcal{H}_K(u^J[1 : L_J]) \end{bmatrix} \quad (36)$$

has full row rank.

Theorem 2 (Modified from Theorem 1 of [32]): Suppose linear time-invariant system (13) is controllable, and assume that the input trajectories $u^j[1 : L_j]$ are collectively persistently exciting of order $n_x + T$. Then every length T input/output trajectory of (13) can be expressed as

$$\begin{bmatrix} \bar{u}[1 : T] \\ \bar{y}[1 : T] \end{bmatrix} = \begin{bmatrix} U^{\mathrm{p}} \\ Y^{\mathrm{p}} \\ U^{\mathrm{f}} \\ Y^{\mathrm{f}} \end{bmatrix} g, \quad (37)$$

where the vector $g \in \mathbb{R}^N$ is in the image space of the matrix $[U^{\mathrm{p}\top}\ Y^{\mathrm{p}\top}\ U^{\mathrm{f}\top}\ Y^{\mathrm{f}\top}]^\top$ and $N = \sum_{j=1}^{J} L_j - JT + J$.

Based on this theorem, given past input u^{p} and past output y^{p} and the prediction of the future input $\hat{u}^{\mathrm{f}}$, we can predict the future output $\hat{y}^{\mathrm{f}}$ by solving the optimization problem

$$\begin{aligned} \min_{g, \hat{y}^{\mathrm{f}}} \ & \frac{1}{2} \|g\|_2^2 \\ \text{s.t.} \ & \begin{bmatrix} U^{\mathrm{p}} \\ Y^{\mathrm{p}} \\ U^{\mathrm{f}} \\ Y^{\mathrm{f}} \end{bmatrix} g = \begin{bmatrix} u^{\mathrm{p}} \\ y^{\mathrm{p}} \\ \hat{u}^{\mathrm{f}} \\ \hat{y}^{\mathrm{f}} \end{bmatrix}, \end{aligned} \quad (38)$$

where the hat symbol $\hat{\cdot}$ refers to the predicted future input and output. Let $\xi := [u^{\mathrm{p}\top}\ y^{\mathrm{p}\top}\ \hat{u}^{\mathrm{f}\top}]^\top$ and $\Xi := [U^{\mathrm{p}\top}\ Y^{\mathrm{p}\top}\ U^{\mathrm{f}\top}]^\top$. Then the optimization becomes

$$\begin{aligned} \min_{g, \hat{y}^{\mathrm{f}}} \ & \frac{1}{2} \|g\|_2^2 \\ \text{s.t.} \ & \begin{bmatrix} \Xi \\ Y^{\mathrm{f}} \end{bmatrix} g = \begin{bmatrix} \xi \\ \hat{y}^{\mathrm{f}} \end{bmatrix}. \end{aligned} \quad (39)$$

Let λ_1 and λ_2 be the Lagrange multipliers with respect to the equality constraints $\Xi g = \xi$ and $Y^{\mathrm{f}} g = \hat{y}^{\mathrm{f}}$, respectively. Then the Karush-Kuhn-Tucker (KKT) conditions are

$$\begin{aligned} g + \Xi^\top \lambda_1 + Y^{\mathrm{f}\top} \lambda_2 &= 0, \\ \lambda_2 &= 0, \end{aligned}$$

$$\Xi g = \xi, \qquad Y^{\mathrm{f}} g = \hat{y}^{\mathrm{f}}. \tag{40}$$

Therefore

$$\Xi\Xi^\top \lambda_1 + \xi = 0, \tag{41}$$

and

$$g = -\Xi^\top \lambda_1 = \Xi^\top \left(\Xi\Xi^\top\right)^\dagger \xi = \Xi^\dagger \xi, \tag{42}$$

is a solution to the KKT condition. Then combining $\hat{y}^{\mathrm{f}} = Y^{\mathrm{f}} g$ and $g = \Xi^\dagger \xi$ gives the prediction

$$\hat{y}^{\mathrm{f}}[k] = Y^{\mathrm{f}} \Xi^\dagger \xi[k] = Y^{\mathrm{f}} \begin{bmatrix} U^{\mathrm{p}} \\ Y^{\mathrm{p}} \\ U^{\mathrm{f}} \end{bmatrix}^\dagger \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \\ \hat{u}^{\mathrm{f}}[k] \end{bmatrix}, \tag{43}$$

[33] which is consistent with the form of the multi-step predictor (31).

B. Autoregressive Prediction

The input-output model predicts the future output y^{f} based on the past input output data u^{p}, y^{p}, as well as the future input u^{f}. In control tasks, the future input u^{f} is usually determined by the control policy, which enables active control of the system output y^{f} [20]. However, for the prediction task, the future input u^{f} (i.e., the future speed of the head vehicle) is unknown. As illustrated in Fig. 2, the task is split into two sub-tasks:

(i) Given the past input u^{p}, predict the future input $\hat{u}^{\mathrm{f}}$.
(ii) Given the past input u^{p}, past output y^{p} and predicted future input $\hat{u}^{\mathrm{f}}$, predict the future output y^{f}.

For task (ii) we simply apply the input-output approach described in Sec. III-B and Sec. IV-A. For task (i), one can use the constant prediction $\hat{u}[k+i] = u[k]$ for $i = 1, \ldots, T^{\mathrm{f}}$ as a baseline. On the other hand, one can assume the input data itself is an autoregressive process. In other words, we would like to find the linear relationship between u^{p} and u^{f}:

$$u^{\mathrm{f}}[k] = \Psi u^{\mathrm{p}}[k]. \tag{44}$$

We refer to this as the *autoregressive model*.

The data-based solution operator from (43) reads as

$$\hat{\Phi} = Y^{\mathrm{f}} \Xi^\dagger = Y^{\mathrm{f}} \begin{bmatrix} U^{\mathrm{p}} \\ Y^{\mathrm{p}} \\ U^{\mathrm{f}} \end{bmatrix}^\dagger. \tag{45}$$

Analogously, the solution operator of the autoregressive model (44) can be obtained from data as

$$\hat{\Psi} = U^{\mathrm{f}} U^{\mathrm{p}\dagger}. \tag{46}$$

The resulting prediction is

$$\begin{aligned} \hat{y}^{\mathrm{f}}[k] &= \hat{\Phi} \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \\ \hat{u}^{\mathrm{f}}[k] \end{bmatrix} = \hat{\Phi} \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \\ \hat{\Psi} u^{\mathrm{p}}[k] \end{bmatrix} = \hat{\Phi} \begin{bmatrix} I & 0 \\ 0 & I \\ \hat{\Psi} & 0 \end{bmatrix} \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \end{bmatrix} \\ &= \underbrace{Y^{\mathrm{f}} \begin{bmatrix} U^{\mathrm{p}} \\ Y^{\mathrm{p}} \\ U^{\mathrm{f}} \end{bmatrix}^\dagger}_{\Xi^\dagger} \begin{bmatrix} I & 0 \\ 0 & I \\ U^{\mathrm{f}} U^{\mathrm{p}\dagger} & 0 \end{bmatrix} \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \end{bmatrix} = \Gamma \begin{bmatrix} u^{\mathrm{p}}[k] \\ y^{\mathrm{p}}[k] \end{bmatrix}, \end{aligned} \tag{47}$$

where Γ is referred to as the autoregressive operator. Given the past input u^{p} and past output y^{p}, the autoregressive model predicts the future output $\hat{y}^{\mathrm{f}}$ via a linear map.

C. Iterative Adaptive Prediction

As the dynamics of the system may change over time, the autoregressive operator needs to be updated online to adapt to the changing environment. We apply recursive least squares (RLS) algorithm to iteratively update the model.

At time step k, before receiving new data, we already have data up to time $k-1$. We can organize the data as U^{p}_{k-1}, Y^{p}_{k-1}, U^{f}_{k-1}, Y^{f}_{k-1}. According to (47), the autoregressive operator at time $k-1$ is obtained as

$$\begin{aligned} \Gamma_{k-1} &= \hat{\Phi}_{k-1} \begin{bmatrix} I & 0 \\ 0 & I \\ \hat{\Psi}_{k-1} & 0 \end{bmatrix} \\ &= Y^{\mathrm{f}}_{k-1} \underbrace{\begin{bmatrix} U^{\mathrm{p}}_{k-1} \\ Y^{\mathrm{p}}_{k-1} \\ U^{\mathrm{f}}_{k-1} \end{bmatrix}^\dagger}_{\Xi^\dagger_{k-1}} \begin{bmatrix} I & 0 \\ 0 & I \\ U^{\mathrm{f}}_{k-1} U^{\mathrm{p}\dagger}_{k-1} & 0 \end{bmatrix}. \end{aligned} \tag{48}$$

With new data $u^{\mathrm{p}}[k]$, $y^{\mathrm{p}}[k]$, $u^{\mathrm{f}}[k]$, $y^{\mathrm{f}}[k]$ at time k, the data matrices are updated as

$$\begin{aligned} U^{\mathrm{p}}_k &= \begin{bmatrix} U^{\mathrm{p}}_{k-1} & u^{\mathrm{p}}[k] \end{bmatrix}, \quad Y^{\mathrm{p}}_k = \begin{bmatrix} Y^{\mathrm{p}}_{k-1} & y^{\mathrm{p}}[k] \end{bmatrix}, \\ U^{\mathrm{f}}_k &= \begin{bmatrix} U^{\mathrm{f}}_{k-1} & u^{\mathrm{f}}[k] \end{bmatrix}, \quad Y^{\mathrm{f}}_k = \begin{bmatrix} Y^{\mathrm{f}}_{k-1} & y^{\mathrm{f}}[k] \end{bmatrix}, \\ \Xi_k &= \begin{bmatrix} \Xi_{k-1} & \xi[k] \end{bmatrix}. \end{aligned} \tag{49}$$

We would like to update $\hat{\Phi}_{k-1}$ and $\hat{\Psi}_{k-1}$ to $\hat{\Phi}_k$ and $\hat{\Psi}_k$, respectively. Observe that

$$\begin{aligned} \hat{\Phi}_k &= Y^{\mathrm{f}}_k \Xi^\dagger_k = Y^{\mathrm{f}}_k \Xi^\top_k \left(\Xi_k \Xi^\top_k\right)^\dagger, \\ \hat{\Psi}_k &= U^{\mathrm{f}}_k U^{\mathrm{p}\dagger}_k = U^{\mathrm{f}}_k U^{\mathrm{p}\top}_k \left(U^{\mathrm{p}}_k U^{\mathrm{p}\top}_k\right)^\dagger. \end{aligned} \tag{50}$$

Let $P_k = \left(\Xi_k \Xi^\top_k\right)^\dagger$, using the Sherman-Morrison formula, we have

$$\begin{aligned} P_k &= \left(P_{k-1} + \xi[k]\xi[k]^\top\right)^\dagger \\ &= P_{k-1} - \frac{P_{k-1}\xi[k]\xi[k]^\top P_{k-1}}{1 + \xi[k]^\top P_{k-1} \xi[k]}. \end{aligned} \tag{51}$$

Therefore, the update rule for $\hat{\Phi}$ is

$$\hat{\Phi}_k = Y^{\mathrm{f}}_k \Xi^\top_k P_k = \left(Y^{\mathrm{f}}_{k-1} \Xi^\top_{k-1} + y^{\mathrm{f}}[k]\xi[k]^\top\right) P_k. \tag{52}$$

Similarly, we can define $Q_k = \left(U^{\mathrm{p}}_k U^{\mathrm{p}\top}_k\right)^\dagger$, yielding

$$\begin{aligned} Q_k &= \left(Q_{k-1} + u^{\mathrm{p}}[k] u^{\mathrm{p}}[k]^\top\right)^\dagger \\ &= Q_{k-1} - \frac{Q_{k-1} u^{\mathrm{p}}[k] u^{\mathrm{p}}[k]^\top Q_{k-1}}{1 + u^{\mathrm{p}}[k]^\top Q_{k-1} u^{\mathrm{p}}[k]}, \end{aligned} \tag{53}$$

and update $\hat{\Psi}$ as

$$\hat{\Psi}_k = U^{\mathrm{f}}_k U^{\mathrm{p}\top}_k Q_k = \left(U^{\mathrm{f}}_{k-1} U^{\mathrm{p}\top}_{k-1} + u^{\mathrm{f}}[k] u^{\mathrm{p}}[k]^\top\right) Q_k. \tag{54}$$

These update rules enable efficient online learning of the autoregressive model.

V. Memory Sketching

In (43), we have $\Xi \in \mathbb{R}^{M \times N}$ with

$$\begin{aligned} M &= T^{\mathrm{p}}(n_u + n_y) + T^{\mathrm{f}} n_u = T n_u + T^{\mathrm{p}} n_y, \\ N &= \sum_{i=1}^{J} L_i - JT + J, \end{aligned} \tag{55}$$

where n_u, n_y are the dimensions of input u and output y, respectively, see (32) and (33). When T is large, the computation of $\Xi^\dagger$ is expensive. We propose to use memory sketching to reduce the computation cost. To resolve this issue we project the original data Ξ and Y^{f} to a lower dimensional space using the projection matrix $\Pi \in \mathbb{R}^{N\times S}$ as follows. Here, M is the regressor dimension determined by the selected horizons and signal dimensions, while N is the number of available data columns and increases as new data arrives.

$$\tilde{Y}^{\mathrm{f}} = Y^{\mathrm{f}}\Pi, \quad \tilde{\Xi} = \Xi\Pi, \tag{56}$$

where $\tilde{\cdot}$ refers to the sketched data. Then the predictor (47) is estimated from the sketched data using the solution operators

$$\tilde{\Phi} = \tilde{Y}^{\mathrm{f}}\tilde{\Xi}^\dagger = \tilde{Y}^{\mathrm{f}}\begin{bmatrix}\tilde{U}^{\mathrm{p}}\\ \tilde{Y}^{\mathrm{p}}\\ \tilde{U}^{\mathrm{f}}\end{bmatrix}^\dagger, \quad \tilde{\Psi} = \tilde{U}^{\mathrm{f}}\tilde{U}^{\mathrm{p}\dagger}, \tag{57}$$

see (45) and (46).

Essentially, we are trying to compress the data into a lower dimensional space without losing too much information. One example is the window sketch

$$\Pi^{\mathrm{w}} = \begin{bmatrix}0_{(N-S)\times S}\\ I_{S\times S}\end{bmatrix}, \tag{58}$$

which keeps the last S columns of the original data matrix. However, this method heavily relies on the quality of the recent data: if the most recent data is not persistently exciting, we need to consider longer history. Instead, we can sample the projection matrix Π from a Gaussian distribution and use the following lemma to guarantee the quality of the projection.

Lemma 1: [Johnson Lindenstrauss (JL) Lemma] For any $0 < \epsilon, \delta < 1$, let $S = O\left(\frac{1}{\epsilon^2}\log\frac{1}{\delta}\right)$, and Π be a random matrix with i.i.d. Gaussian entries, $\Pi_{ij} \sim \mathcal{N}(0, 1/S)$. Then for any $z \in \mathbb{R}^N$, with probability $1-\delta$ we have

$$(1-\epsilon)\|z\|_2^2 \le \|\Pi z\|_2^2 \le (1+\epsilon)\|z\|_2^2. \tag{59}$$

This is also referred to as the *restricted isometry property*. Based on the JL Lemma, we have the following error bound:

Theorem 3: Let ϵ, δ, S, Π defined as in Lemma 1, then with probability at least $1-\delta$ we have

$$\left\|\tilde{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \le \frac{1+\epsilon}{1-\epsilon}\left\|\hat{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2. \tag{60}$$

where $\|\cdot\|_{\mathrm{F}}$ is Frobenius norm.

Proof: With probability at least $1-\delta$ we have

$$\begin{aligned}\left\|\tilde{\Phi}\tilde{\Xi} - \tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^2 &\le \left\|\hat{\Phi}\tilde{\Xi} - \tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^2 = \left\|(\hat{\Phi}\Xi - Y^{\mathrm{f}})\Pi\right\|_{\mathrm{F}}^2\\ &\le (1+\epsilon)\left\|\hat{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2.\end{aligned} \tag{61}$$

On the other hand

$$\left\|\tilde{\Phi}\tilde{\Xi} - \tilde{Y}^{\mathrm{f}}\right\|_{\mathrm{F}}^2 = \left\|(\tilde{\Phi}\Xi - Y^{\mathrm{f}})\Pi\right\|_{\mathrm{F}}^2 \ge (1-\epsilon)\left\|\tilde{\Phi}\Xi - Y^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \tag{62}$$

□

Similarly, we have

$$\left\|\tilde{\Psi}U^{\mathrm{p}} - U^{\mathrm{f}}\right\|_{\mathrm{F}}^2 \le \frac{1+\epsilon}{1-\epsilon}\left\|\hat{\Psi}U^{\mathrm{p}} - U^{\mathrm{f}}\right\|_{\mathrm{F}}^2. \tag{63}$$

Furthermore, since all elements of Π are i.i.d., the sketches can be computed incrementally, which is suitable for online learning. Namely, consider that at time step $k-1$ we have

$$\tilde{\Xi}_{k-1} = \Xi_{k-1}\Pi = \begin{bmatrix}\xi[1] & \xi[2] & \cdots & \xi[k-1]\end{bmatrix}\begin{bmatrix}\pi_1\\ \pi_2\\ \vdots\\ \pi_{k-1}\end{bmatrix}. \tag{64}$$

When a new data $\xi[k]$ arrives, we can update the sketch as

$$\tilde{\Xi}_k = \tilde{\Xi}_{k-1} + \xi[k]\pi_k^\top. \tag{65}$$

At the same time, new output vector $y^{\mathrm{f}}[k]$ is generated which results in the update

$$\tilde{Y}_k^{\mathrm{f}} = \tilde{Y}_{k-1}^{\mathrm{f}} + y^{\mathrm{f}}[k]\pi_k^\top. \tag{66}$$

Note that the projection vector π_k is independent of the data, but Ξ and Y^{f} shall share the same projection vector.

Remark 1: [computational complexity] We use three dimensions throughout this paper: M (regressor dimension), N (number of historical samples), and S (sketch size). A full-dimension RLS update has per-step complexity $\mathcal{O}(M^2)$ and memory $\mathcal{O}(M^2)$. With sketching, the online update in the sketched space has per-step complexity $\mathcal{O}(S^2)$ and memory $\mathcal{O}(S^2)$. Therefore, sketching is computationally beneficial when $S < M$, and most effective when $S \ll M$. In that regime, the per-step update cost is reduced by approximately a factor of $(M/S)^2$. For comparison, re-solving batch least squares from the entire history at every step scales as $\mathcal{O}(NM^2 + M^3)$ and thus grows with N.

Remark 2 (packet loss handling): In practical V2X deployments, packet losses may occur that can disrupt the continuous stream of input data. When the input u is unavailable due to packet loss, we employ a simple prediction-reuse strategy:

1) Primary: Query the prediction cache for a previously predicted value at the current timestep.
2) Fallback: If no cached prediction is available, use zero-order hold (the last received value).

This approach leverages the model's own forecasts to maintain continuity without complex interpolation schemes. Importantly, to prevent the model from overfitting to its own predictions, we mark filled values and exclude them from RLS parameter updates. This keeps the approach computationally efficient while maintaining prediction accuracy under realistic communication impairments.

VI. Simulation Results

In this section, we evaluate the performance of the proposed methods on the car-following example. We first show the effect of persistency of excitation on a simplistic scenario. Then we demonstrate the performance of the proposed method on a realistic car-following scenario.

A. Persistency of Excitation

Here we consider a simple car-following example with $L = 2$, where the connected vehicle L is right in front of vehicle 1, c.f. Fig 1. In this case,

$$x = x_1, \quad u = \tilde{v}_2, \quad y = \tilde{v}_1, \tag{67}$$

TABLE II
PARAMETERS FOR OVM MODEL

α [1/s]	β [1/s]	κ [1/s]	r	Δt [s]	T^{p}	T^{f}
0.4	0.5	0.6	80	0.1	83	50

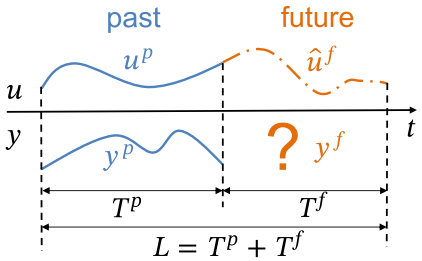


Fig. 2. Data-driven prediction: (i) predict future input $\hat{u}^{\mathrm{f}}$ based on past input u^{p}; (ii) predict future output y^{f} based on past input u^{p}, past output y^{p} and predicted future input $\hat{u}^{\mathrm{f}}$.

and the system dynamics (13) becomes

$$x[k+1] = A_1 x[k] + B_1 u[k],$$
$$y[k] = C_1 x[k]. \tag{68}$$

where A_1, B_1, C_1 are defined in (10). Here we use the parameters shown in Table II.

Note that in Theorem 2, the input trajectories $u^j[1:L_j]$, $j = 1,\ldots,J$ are expected to be collectively persistently exciting of order $n_x + T$. We investigate the effect of different input trajectories on the prediction performance. We consider three types of input trajectories: zero input, sine input, and Matérn input. The zero input is kept 0 for all time. The sine input follows sinusoidal function $u(t) = A\sin(\omega t + \phi)$ where A, ω and ϕ are scalars. The Matérn input is sampled from Gaussian process with Matérn kernel

$$C_\nu(\tau) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu}\frac{\tau}{l}\right)^\nu K_\nu \left(\sqrt{2\nu}\frac{\tau}{l}\right), \tag{69}$$

which gives the correlation between $u(t)$ and $u(t+\tau)$ for any t. Here Γ is the gamma function and K_ν is the modified Bessel function of the second kind. We set $\nu = 3/2$, which ensures the signal is differentiable [34]. The initial condition is generated by the normal distribution $x[0] \sim \mathcal{N}(0, I_{n_x})$ where I_{n_x} is the n_x-dimensional identity matrix.

We consider the prediction performance with different sketch sizes S ranging from 1 to 1000, where 1000 is the total length of data, therefore this is equivalent to full data prediction. The prediction error is measured by the root-mean-square error (RMSE) between the predicted output and the true output. The prediction horizon is $T^{\mathrm{f}} = 10$, and the past horizon is $T^{\mathrm{p}} = 8$, which correspond to 1 [s] and 0.8 [s] respectively, since the discretization step $\Delta t = 0.1$ [s] is used.

For zero input (see Fig. 3) and for sine input (see Fig. 4), the dataset is not persistently excited. The ranks of Hankel matrices $\mathcal{H}_{n_x+T}(u)$ are 0 and 3 respectively. Therefore, although the training error can achieve 0, when we compare the estimated solution operator $\hat{\Phi}$ given in (45) with the ground truth Φ given in (31), the error remains positive. However, when the input signal is generated by a Gaussian process with Matérn kernel (see Fig. 5), the dataset is persistently excited, the error of solution operator approaches zero as the sketch size increases. Regardless of the input signal, the Gaussian sketch method can achieve the same prediction performance as the full data method despite the small sketch size.

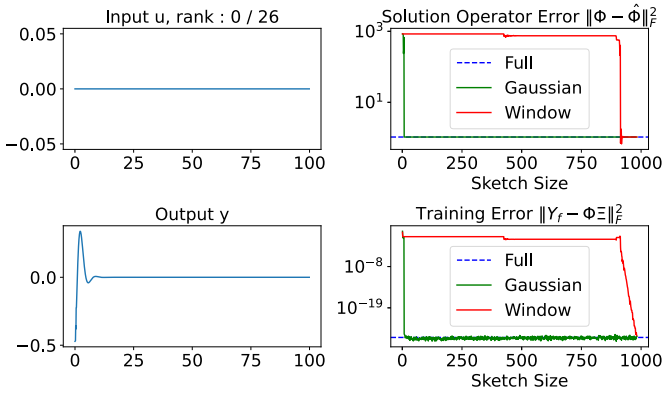


Fig. 3. Prediction performance with zero input. The rank of Hankel matrix $\mathcal{H}_{n_x+T}(u)$ is 0. The ground truth dynamics cannot be fully captured. The Gaussian sketch method can achieve the same prediction performance as the full data method with small sketch size.

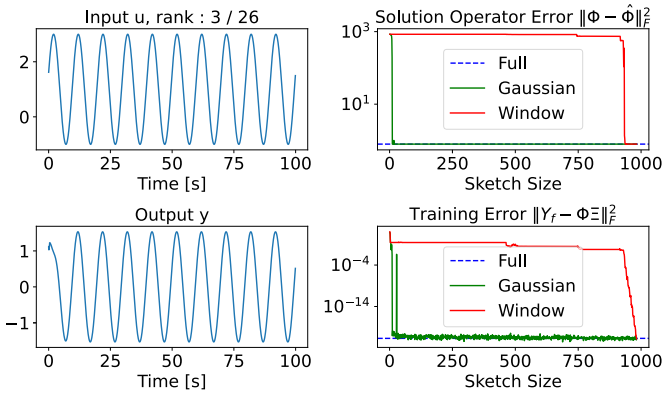


Fig. 4. Prediction performance with sine input. The rank of Hankel matrix $\mathcal{H}_{n_x+T}(u)$ is 3. The ground truth dynamics cannot be fully captured. The Gaussian sketch method can achieve the same prediction performance as the full data method with small sketch size.

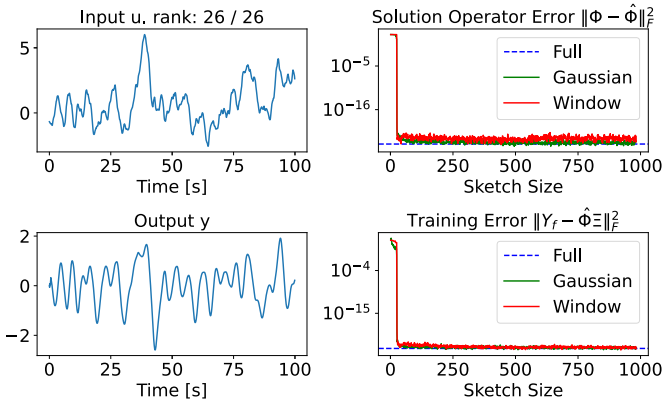


Fig. 5. Prediction performance with Matérn input. The system is persistently excited so the ground truth dynamics is fully captured. Both the Gaussian sketch method and the window sketch method can achieve the same prediction performance as the full data method with small sketch size.

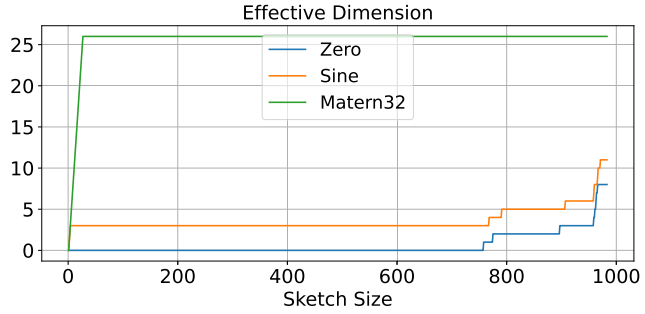


Fig. 6. Effective dimension for different sketch sizes. The zero signal and the sine signal fail to match the persistency of excitation condition, while the Matérn signal satisfies the condition even for small sketch sizes.

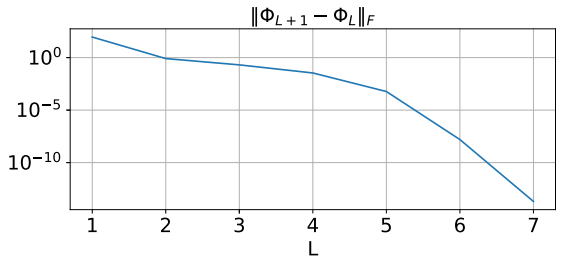


Fig. 7. Difference of multi-step predictor matrix describing the head-to-tail dynamics when changing the head vehicle L.

To illustrate the effect of persistency of excitation, in Fig. 6 we plot the effective dimension of matrices

$$\begin{bmatrix} \mathcal{H}_T(u[:-S]) \\ \mathcal{H}_T(y[:-S]) \end{bmatrix}, \tag{70}$$

for different sketch sizes S, where $u[:-S]$ and $y[:-S]$ denote the last S columns of trajectory u and y. The effective dimension is defined as the number of singular values that are larger than a threshold, chosen to be 10^{-5}. The input-output trajectory fails to cover the full space when the input signal is a zero or a sine signal, while Matérn signal enables us to capture the full space even for small sketch size.

B. Prediction With V2X Communication

Here we investigate a car-following scenario where the ego vehicle (CAV) is following a chain of human-driven vehicles as detailed in Sec. II. We assume that the head vehicle is a connected vehicle and it performs stop-and-go motion due to the traffic lights. The trajectory data used in this section are synthetic and generated from a Gaussian process, see Figs. 8 and 9. Vehicles between the head vehicle and the ego vehicle carry out car following.

First, to gain an intuitive understanding of the problem, we explicitly calculate the solution operator Φ_L given in (31) for head-to-tail dynamics, where L indicates the head vehicle. To simplify the example, we assume all the drivers share the same OVM parameters listed in Table II. To compare the head-to-tail dynamics for varying head vehicle, we calculate $\left\|\Phi_{L+1}-\Phi_L\right\|_{\mathrm{F}}$, as is shown in Fig. 7. This shows that the multi-step predictor is sensitive to L when L is small, while the sensitivity decreases exponentially as L increases. That is, the head-to-tail dynamics is more sensitive to L when the head vehicle is closer to the ego vehicle.

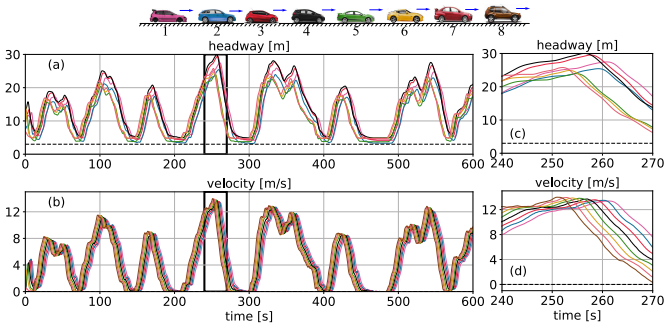


Fig. 8. Synthetic trajectories generated from a Gaussian process used to warm-start the memory sketch. This data represents stop-and-go traffic dynamics with frequent speed variations. The headway and the speed of all the vehicles in the lane are plotted. For sketching we only use the speed data v_8 of the brown vehicle and the speed data v_1 of the pink vehicle as input and output. (a) Headway of all the vehicles. (b) Speed of all the vehicles. (c) Zoom-in of the headways between time 240 [s] and 270 [s]. (d) Zoom-in of the speeds between time 240 [s] and 270 [s].

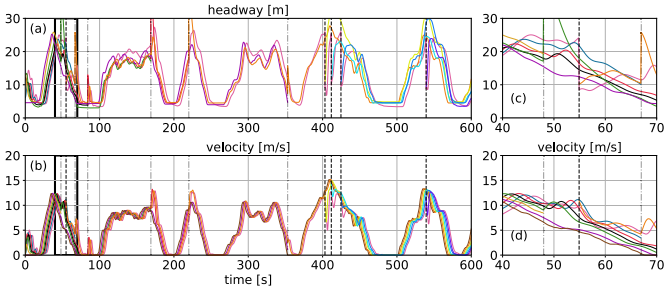


Fig. 9. Synthetic test trajectories representing different traffic conditions. The gray vertical dash-dotted lines indicate the times when a vehicle exits the lane, while the black vertical dashed line indicates when a vehicle merges into the lane. (a) Headway of all the vehicles. (b) Speed of all the vehicles. (c) Zoom-in of the headways between time 40 [s] and 70 [s]. (d) Zoom-in of the speeds between time 40 [s] and 70 [s].

We initialize the memory sketch with some "warm-start" data shown in Fig. 8 where the headway and the speed of all the vehicles in the lane are plotted. We only use the input data v_8, the speed of the brown head vehicle, and the output data v_1, the speed of the pink vehicle right in front of the ego vehicle. With window sketching, we only use the last S data points of the Hankelized input-output data as in (70). The Gaussian sketching is obtained by projecting the Hankelized input-output data to a lower dimensional space with a Gaussian projection matrix of sketch size S.

Then we evaluate the algorithms on the test data shown in Fig. 9. We consider that vehicles merge into and exit of the lane, which leads to changes in the head-to-tail dynamics. The gray vertical dash-dotted lines indicate when a vehicle exits the lane, while black vertical dashed lines indicate when a vehicle merges into the lane. We compare the performance of autoregressive predictions when using the full data, the window sketch method and the Gaussian sketch method. The prediction error is measured by the root-mean-square error (RMSE) $\left\|y^{\mathrm{f}} - \hat{y}^{\mathrm{f}}\right\|_2$.

A direct consequence of the stop-and-go input signal is that the speed of the head vehicle remains constant for long time

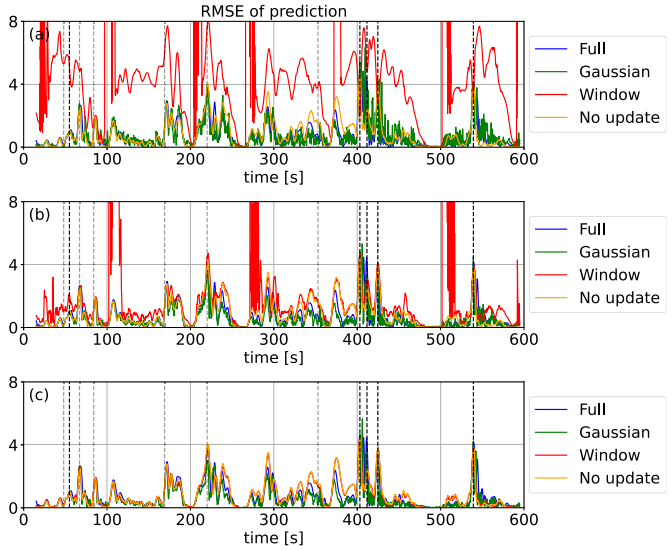


Fig. 10. Comparison of autoregressive prediction without online update of the predictor and with adaptive predictors while utilizing the full data, the window sketch and the Gaussian sketch. (a) Sketch size $S = 50$. (b) Sketch size $S = 200$. (c) Sketch size $S = 400$.

periods. For the window sketch method, we keep dropping old data and completely rely on the prediction on the new data. The constant speed signal is not persistently exciting, which prevents the data sketch from capturing the dynamics of the system. On the other hand, the restricted isometry property (59) of the Gaussian sketch method compresses the data without losing too much information. Fig. 10 shows that, when the head vehicle's speed stays at 0 for a while, there is a large jump in the prediction error for the window sketch, while the prediction error for the Gaussian sketch remains consistent.

The three panels in Fig. 10 compare the prediction error for sketch sizes $S = 50$, 200 and 400, which correspond to 5, 20, and 40 seconds of historical data, respectively. For sketch size $S = 50$, the window sketch method performs poorly. The prediction error is much larger than the full data prediction. The Gaussian sketch method performs better than window sketch, but still worse than full data prediction. For sketch size $S = 200$, the prediction error of the window sketch is reduced, but it is still larger than the full data prediction. The prediction error of the Gaussian sketch is also reduced, and is close to the full data prediction. For sketch size $S = 400$, the prediction errors of the window sketch and the Gaussian sketch are both close to the full data prediction, and the Gaussian sketch performs slightly better than window sketch. The no update curve in Fig. 10 refers to the case when predictions obtained from the warm start are applied without adapting them based on the incoming data. One may observe that this lack of adaptation significantly increases the error compared to the case when using Gaussian updates.

In Fig. 11, we compare the baseline of predictions obtained without online adaptation of the predictor to the cases with Gaussian adaptation. The baseline method will only use the data matrix obtained in the warm-start phase. We highlight the benefits of online adaptation by comparing the prediction error of adaptive and non-adaptive method at each time step.

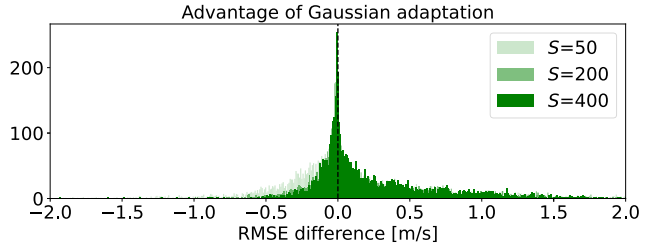


Fig. 11. Comparison between prediction errors without online adaptation and with Gaussian adaptation for different sketch sizes.

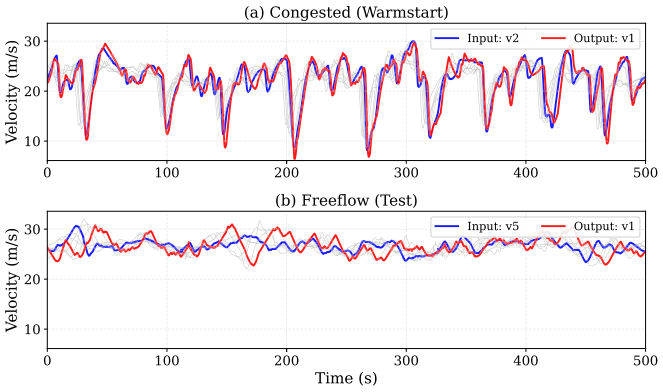


Fig. 12. Velocity profiles from the dataset presented in [30] and [35]. (top) Congested traffic condition used for training with vehicle 2 as input (blue). (bottom) Freeflow traffic condition used for testing with vehicle 5 as input (blue). The output signal is always vehicle 1 (red), the vehicle immediately ahead of the CAV. Gray lines show intermediate vehicles..

On average, adaptation with Gaussian sketch results in 8% improvement in the prediction error when $S = 50$, 30.0% improvement when $S = 200$, and 31.6% improvement when $S = 400$. There is a large improvement when we increase the sketch size from small size ($S = 50$) to medium size ($S = 200$), but the improvement saturates as the sketch size increases.

C. Validation on Experimental Data

To evaluate the robustness of our proposed method under realistic conditions, we conducted comprehensive experiments using real-world traffic data [30], [35]. This dataset was collected during experimental validation of connected automated vehicle designs on public highways, capturing naturalistic human driving behaviors and V2X communication patterns.

As depicted in Fig. 12, the dataset contains multiple driving conditions, including

- Congested: Frequent, large-amplitude speed fluctuations, characteristic of congested urban highways.
- Freeflow: Moderate speed variations, representative of sparse freeway traffic.

In the following experiments, we investigate input distribution shift adaptation, limited training data scenarios, and packet loss robustness.

Experiment A: Input Distribution Shift. To create a challenging input distribution shift scenario, we train on congested data using vehicle 2 as input, and test on freeflow data

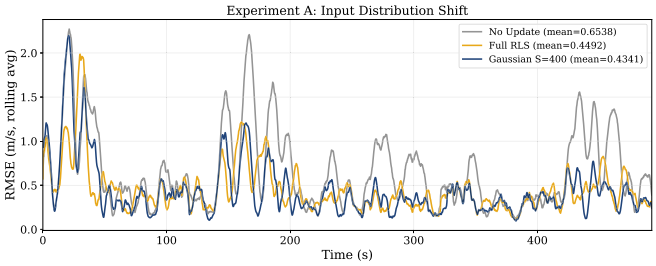


Fig. 13. Prediction RMSE over time, using a rolling average with a 50-step window. The Gaussian sketch method ($S = 400$) maintains consistently lower error than both full RLS and the non-adaptive baseline.

TABLE III
EXPERIMENT A: INPUT DISTRIBUTION SHIFT

Method	RMSE [m/s]	Improvement
No Update (baseline)	0.652	—
Full RLS	0.446	+31.6%
Gaussian $S = 100$	0.450	+31.0%
Gaussian $S = 200$	0.450	+31.0%
Gaussian $S = 400$	**0.428**	**+34.4%**

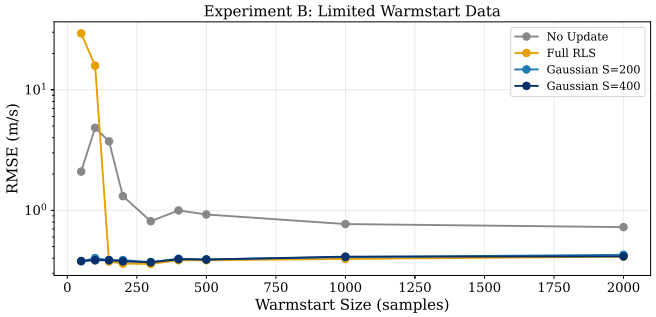


Fig. 14. RMSE as a function of the warmstart size (log scale). The Gaussian sketch provides massive benefits when initial data is scarce and remains beneficial even with larger warmstart sizes.

using vehicle 5 as input. The output is always vehicle 1, the vehicle immediately in front of the CAV. The training input (vehicle 2 in congested) has a standard deviation of 4.27 m/s, while the test input (vehicle 5 in freeflow) has a standard deviation of 1.18 m/s. This input distribution shift tests the model's ability to adapt when the relationship between input and output changes. As shown in Table III and Fig. 13, the Gaussian sketch method with $S = 400$ achieves an RMSE of 0.428 m/s, outperforming the full RLS method (0.446 m/s) by 4.0% and the non-adaptive baseline (0.652 m/s) by 34.4%. This demonstrates that the regularization effect of sketching enhances generalization when the input distribution changes between training and testing.

Experiment B: Limited Warmstart Data. We investigate the performance when only a small amount of data is available for initialization. As shown in Table IV and Fig. 14, with only 100 samples, the Gaussian sketch method with $S = 400$ achieves an RMSE of 0.384 m/s, massively outperforming the baseline (4.84 m/s) by 92.1%. Even with 2000 warmstart samples, the Gaussian sketch continues to provide 42.8% improvement. Notably, the full RLS becomes "unstable" with limited warmstart data, while Gaussian sketch remains robust due to its implicit regularization.

TABLE IV
EXPERIMENT B: LIMITED WARMSTART DATA

Warmstart	Baseline	Best Gaussian	Improvement
50	2.09 m/s	$S = 200$: 0.378 m/s	+82.0%
100	4.84 m/s	$S = 400$: 0.384 m/s	+92.1%
150	3.74 m/s	$S = 400$: 0.386 m/s	+89.7%
200	1.31 m/s	$S = 400$: 0.377 m/s	+71.2%
300	0.81 m/s	$S = 100$: 0.370 m/s	+54.4%
400	1.00 m/s	$S = 400$: 0.393 m/s	+60.6%
500	0.92 m/s	$S = 400$: 0.390 m/s	+57.7%
1000	0.77 m/s	$S = 100$: 0.407 m/s	+47.1%
2000	0.73 m/s	$S = 400$: 0.414 m/s	+42.8%

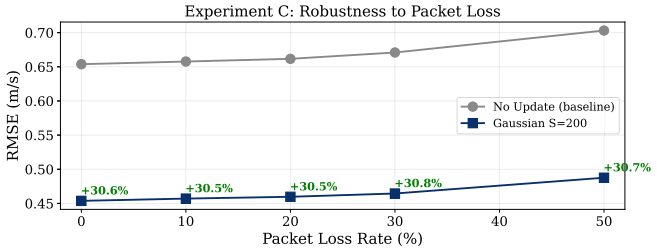


Fig. 15. RMSE as a function of the packet loss rate. The Gaussian sketch method with $S = 200$ maintains 30.7% improvement over baseline across 0–50% loss rates, demonstrating robustness to V2X communication impairments.

TABLE V
EXPERIMENT C: ROBUSTNESS TO PACKET LOSS

Loss Rate	Baseline	Gaussian $S = 200$	Improvement
0%	0.654 m/s	0.454 m/s	+30.6%
10%	0.658 m/s	0.457 m/s	+30.5%
20%	0.662 m/s	0.460 m/s	+30.5%
30%	0.671 m/s	0.464 m/s	+30.8%
50%	0.703 m/s	0.488 m/s	+30.7%

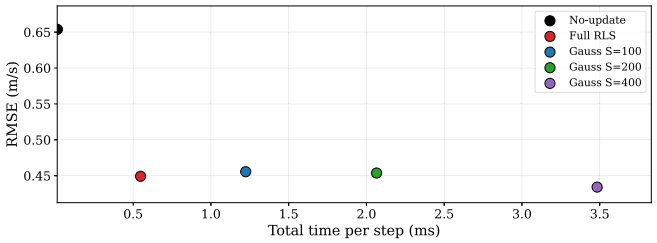


Fig. 16. Time-accuracy tradeoff for different methods. The Gaussian sketch with $S = 400$ achieves the best mean RMSE (0.428 m/s), outperforming both the full RLS (0.446 m/s) and the non-adaptive baseline (0.652 m/s).

TABLE VI
DATA EFFICIENCY AND COMPUTATIONAL COST

Method	Memory	RMSE [m/s]	Std [m/s]	Time [ms]
No Update	$\mathcal{O}(1)$	0.652	0.522	0.01
Full RLS	$\mathcal{O}(M^2)$	0.446	**0.350**	0.53
Gaussian $S = 100$	$\mathcal{O}(S^2)$	0.450	0.383	1.22
Gaussian $S = 200$	$\mathcal{O}(S^2)$	0.450	0.367	2.06
Gaussian $S = 400$	$\mathcal{O}(S^2)$	**0.428**	0.379	3.55
Gaussian $S = 800$	$\mathcal{O}(S^2)$	0.454	0.403	5.72

Experiment C: Robustness to Packet Loss. To address concerns regarding V2X communication reliability, we evaluate the method under varying packet loss rates (0–50%) while using the challenging input distribution shift scenario. As shown in Table V and Fig. 15, the Gaussian sketch method with $S = 200$ maintains approximately 30.5–30.8% improvement over the baseline across all packet loss rates. The RMSE increases moderately from 0.454 m/s at 0% loss to 0.488 m/s at 50% loss, demonstrating that memory sketching is robust to realistic V2X communication impairments.

Finally, Table VI and Fig. 16 summarize the trade-offs between different methods. While memory sketching reduces per-step update cost from $\mathcal{O}(M^2)$ to $\mathcal{O}(S^2)$, Fig. 16 shows that full RLS is also computationally feasible for the tested setting (0.53 ms/step).

VII. Conclusion and Discussion

In this paper, we proposed a data-driven prediction framework for vehicle speed prediction under sparse V2X connectivity, addressing practical challenges including input distribution shifts, limited training data, and packet loss. Starting from classical state-space form of car-following models, we derived input-output models that capture head-to-tail dynamics without requiring knowledge of intermediate vehicle states. We developed a Gaussian sketching algorithm that enables online adaptation, reducing per-step complexity compared to full-dimension recursive least squares algorithm.

Through comprehensive experiments with real-world traffic data, we demonstrated that the proposed method achieves up to 34.4% improvement in prediction accuracy over non-adaptive baselines under input distribution shift conditions, and notably outperforms full RLS by 4.0%. The method provides massive benefits (92.1% improvement) with limited warmstart data (100 samples), where full RLS becomes numerically unstable. Critically, we showed that approximately 30.7% improvement is maintained even under 50% V2X packet loss, addressing a key concern for practical deployment.

From a control perspective, improved speed prediction enables connected cruise control to anticipate braking events and enhance string stability [10]. The consistent gains under input distribution shift, limited warmstart data, and packet loss indicate that the method can support CCC in realistic and degraded communication environments [11]. Future work includes extending this method to multi-vehicle cooperative scenarios and integrating the prediction framework with control for closed-loop performance optimization.

References

[1] G. Orosz, J. I. Ge, C. R. He, S. S. Avedisov, W. B. Qin, and L. Zhang, "Seeing beyond the line of site–controlling connected automated vehicles," *Mech. Eng.*, vol. 139, no. 12, pp. S8–S12, Dec. 2017.

[2] J. Guanetti, Y. Kim, and F. Borrelli, "Control of connected and automated vehicles: State of the art and future challenges," *Annu. Rev. Control*, vol. 45, pp. 18–40, Jun. 2018.

[3] T. Ersal et al., "Connected and automated road vehicles: State of the art and future challenges," *Vehicle Syst. Dyn.*, vol. 58, no. 5, pp. 672–704, May 2020.

[4] S. E. Shladover, C. Nowakowski, X.-Y. Lu, and R. Ferlis, "Cooperative adaptive cruise control: Definitions and operating concepts," *Transp. Res. Record, J. Transp. Res. Board*, vol. 2489, no. 1, pp. 145–152, Jan. 2015.

[5] Z. Wang, G. Wu, and M. Barth, "A review on cooperative adaptive cruise control (CACC) systems: Architectures, controls, and applications," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, 2018, pp. 2884–2891.

[6] H. Liu, S. E. Shladover, X.-Y. Lu, and X. Kan, "Freeway vehicle fuel efficiency improvement via cooperative adaptive cruise control," *J. Intell. Transp. Syst.*, vol. 25, no. 6, pp. 574–586, Nov. 2021.

[7] Y. Zheng, S. E. Li, K. Li, F. Borrelli, and J. K. Hedrick, "Distributed model predictive control for heterogeneous vehicle platoons under unidirectional topologies," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 3, pp. 899–910, May 2017.

[8] V. Turri, B. Besselink, and K. H. Johansson, "Cooperative look-ahead control for fuel-efficient and safe heavy-duty vehicle platooning," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 1, pp. 12–28, Jan. 2017.

[9] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, "String stability for vehicular platoon control: Definitions and analysis methods," *Annu. Rev. Control*, vol. 47, pp. 81–97, Jun. 2019.

[10] G. Orosz, "Connected cruise control: Modelling, delay effects, and nonlinear behaviour," *Vehicle Syst. Dyn.*, vol. 54, no. 8, pp. 1147–1176, Aug. 2016.

[11] M. Shen, R. A. Dollar, T. G. Molnár, C. R. He, A. Vahidi, and G. Orosz, "Energy-efficient reactive and predictive connected cruise control," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 1, pp. 944–957, Jan. 2024.

[12] Y. Sugiyama, "Optimal velocity model for traffic flow," *Comput. Phys. Commun.*, vols. 121–122, pp. 399–401, Sep. 1999.

[13] T. G. Molnar and G. Orosz, "Destroying phantom jams with connectivity and automation: Nonlinear dynamics and control of mixed traffic," *Transp. Sci.*, vol. 58, no. 6, pp. 1319–1334, Nov. 2024.

[14] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 62, no. 2, pp. 1805–1824, Aug. 2000.

[15] G. F. Newell, "A simplified car-following theory: A lower order model," *Transp. Res. B, Methodol.*, vol. 36, no. 3, pp. 195–205, Mar. 2002.

[16] X. Wang et al., "Traffic light optimization with low penetration rate vehicle trajectory data," *Nature Commun.*, vol. 15, no. 1, p. 1306, Feb. 2024.

[17] M. Shen and G. Orosz, "Data-driven predictive connected cruise control," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2023, pp. 1–6.

[18] L. Zhang and G. Orosz, "Motif-based design for connected vehicle systems in presence of heterogeneous connectivity structures and time delays," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1638–1651, Jun. 2016.

[19] I. Markovsky and F. Dörfler, "Behavioral systems theory in data-driven analysis, signal processing, and control," *Annu. Rev. Control*, vol. 52, pp. 42–64, Dec. 2021.

[20] J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control: In the shallows of the DeePC," in *Proc. 18th Eur. Control Conf. (ECC)*, 2019, pp. 307–312.

[21] E. Elokda, J. Coulson, P. N. Beuchat, J. Lygeros, and F. Dörfler, "Data-enabled predictive control for quadcopters," *Int. J. Robust Nonlinear Control*, vol. 31, no. 18, pp. 8916–8936, Dec. 2021.

[22] J. Wang, Y. Zheng, K. Li, and Q. Xu, "DeeP-LCC: Data-EnablEd predictive leading cruise control in mixed traffic flow," *IEEE Trans. Control Syst. Technol.*, vol. 31, no. 6, pp. 2760–2776, Nov. 2023.

[23] K. Zhang, Y. Zheng, C. Shang, and Z. Li, "Dimension reduction for efficient data-enabled predictive control," *IEEE Control Syst. Lett.*, vol. 7, pp. 3277–3282, 2023.

[24] D. P. Woodruff, "Sketching as a tool for numerical linear algebra," *Found. Trends Theor. Comput. Sci.*, vol. 10, nos. 1–2, pp. 1–157, Oct. 2014.

[25] G. Cormode and S. Muthukrishnan, "An improved data stream summary: The count-min sketch and its applications," *J. Algorithms*, vol. 55, no. 1, pp. 58–75, Apr. 2005.

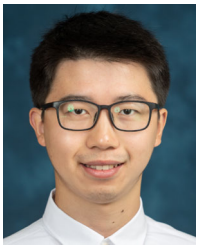
[26] M. Pilanci and M. J. Wainwright, "Randomized sketches of convex programs with sharp guarantees," *IEEE Trans. Inf. Theory*, vol. 61, no. 9, pp. 5096–5115, Sep. 2015.

[27] H. Zhou, A. Zhou, T. Li, D. Chen, S. Peeta, and J. Laval, "Congestion-mitigating MPC design for adaptive cruise control based on Newell's car following model: History outperforms prediction," *Transp. Res. C, Emerg. Technol.*, vol. 142, Sep. 2022, Art. no. 103801.

[28] T. G. Molnar et al., "On-board traffic prediction for connected vehicles: Implementation and experiments on highways," in *Proc. Amer. Control Conf. (ACC)*, Jun. 2022, pp. 1036–1041.

[29] J. I. Ge and G. Orosz, "Connected cruise control among human-driven vehicles: Experiment-based parameter estimation and optimal control design," *Transp. Res. C, Emerg. Technol.*, vol. 95, pp. 445–459, Oct. 2018.

[30] G. Orosz and T. G. Molnár, *Dynamics and Control of Connected Vehicles*. Cham, Switzerland: Springer, 2025.
[31] L. Zhang and G. Orosz, "Consensus and disturbance attenuation in multi-agent chains with nonlinear control and time delays," *Int. J. Robust Nonlinear Control*, vol. 27, no. 5, pp. 781–803, Mar. 2017.
[32] H. J. van Waarde, C. De Persis, M. K. Camlibel, and P. Tesi, "Willems' fundamental lemma for state-space systems and its extension to multiple datasets," *IEEE Control Syst. Lett.*, vol. 4, no. 3, pp. 602–607, Jul. 2020.
[33] I. Markovsky and P. Rapisarda, "Data-driven simulation and control," *Int. J. Control*, vol. 81, no. 12, pp. 1946–1959, Dec. 2008.
[34] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, vol. 2. Cambridge, MA, USA: MIT Press, 2006.
[35] J. I. Ge, S. S. Avedisov, C. R. He, W. B. Qin, M. Sadeghpour, and G. Orosz, "Experimental validation of connected automated vehicle design among human-driven vehicles," *Transp. Res. C, Emerg. Technol.*, vol. 91, pp. 335–352, Jun. 2018.



Minghao Shen (Graduate Student Member, IEEE) received the B.Eng. degree in automotive engineering from Jilin University in 2019. He is currently pursuing the Ph.D. degree in mechanical engineering with the University of Michigan, Ann Arbor, MI, USA. His research interests include stochastic control, machine learning, data-driven control, and their applications in the planning and control of connected and automated vehicles.



Jeff Scruggs (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Virginia Tech, Blacksburg, VA, USA, in 1997 and 1999, respectively, and the Ph.D. degree in applied mechanics from Caltech in 2004. In 2011, he joined the University of Michigan, Ann Arbor, MI, USA, where he is currently a Professor of civil and environmental engineering, as well as electrical and computer engineering. Prior to joining the University of Michigan, he was a Post-Doctoral Researcher at Caltech and the University of California, San Diego, and was a Faculty Member at Duke University from 2007 to 2011. His current research interests include mechanics, vibration, energy, and control.



Gábor Orosz (Senior Member, IEEE) received the M.Sc. degree in engineering physics from Budapest University of Technology, Hungary, in 2002, and the Ph.D. degree in engineering mathematics from the University of Bristol, U.K., in 2006. From 2017 to 2018, he was a Visiting Professor in control and dynamical systems with California Institute of Technology. In 2022, he was a Distinguished Guest Researcher in applied mechanics with Budapest University of Technology, where he was a Fulbright Scholar from 2023 to 2024. He is currently a Professor of mechanical engineering and a Professor of civil and environmental engineering with the University of Michigan. His research interests include nonlinear dynamics and control, time delay systems, machine learning, and data-driven systems with applications to connected and automated vehicles, traffic flow, and biological networks.