Reconstruction of mixed traffic systems at micro and macro scales

By

Yanbing Wang

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Civil and Environmental Engineering

December 16, 2023

Nashville, Tennessee

Approved:

Dan Work, Ph.D.

Maria Laura Delle Monache, Ph.D. (University of California at Berkeley)

Sankaran Mahadevan, Ph.D.

Jonathan Sprinkle, Ph.D.

Ahmad Taha, Ph.D.

ACKNOWLEDGEMENT

Throughout my PhD, I've been incredibly fortunate to have the guidance and support of my advisor, Dr. Dan Work. From him, I've learned a valuable lesson: "dream big and work small," a philosophy that he himself embodies. I had the privilege to witness the creation of the largest traffic testbed, a project meticulously crafted over years of incremental advancements. Countless insightful discussions with Dan during this process have been instrumental in turning my initial concepts into publications and real-world applications. His support and encouragement have granted me the freedom to explore venturesome academic pursuits. I am deeply grateful for his mentorship towards both my academic and personal development.

This endeavour would not have been made possible without the support and genuine feedback from my committee members: Professors Maria Laura Delle Monache, Sankaran Mahadevan, Jonathan Sprinkle, and Ahmad Taha. Each of them has played a pivotal role in shaping the trajectory of my academic growth. Prof. Delle Monache's meticulous proofreading and insightful mathematical guidance advanced my understanding of parameter identifiability. Prof. Mahadevan has been a guiding mentor to me since my very first day at Vanderbilt, from whom I learned a wealth of engineering skills and knowledge. Prof. Taha's infectious positivity and meticulous attention to detail have motivated excellence in my work. I enjoy the many coffee chats with Prof. Sprinkle, whose creative ideas and passion for progress motivated my progress. For their dedication and belief in my potential, I extend my deepest gratitude.

I am grateful to Dr. Janos Sztipanovits who introduced me to the Institute for Software Integrated System at Vanderbilt. Through the seminars and social events at the institute I met a diverse group of faculties and students from Computer Science and Electrical Engineering. The collaborative environment had stimulated many brainstorming discussions which led me to venture into the field of cyber-physical systems, where I found research opportunities to bridge traditional civil engineering with efficient computational techniques.

My PhD years at Vanderbilt are made more colorful with the comradeship of my colleagues and friends. I am always inspired by the dedication from the "Work Lab" folks: Will Barbour, Derek Gloudemans, George Gunter, Yue Hu, Junyi Ji, Matt Nice, Gergeley Zachar, Yuhang Zhang and Zhiyao Zhang. I am also grateful for many undergraduate interns who have worked with me in the past. They all have taught me how to be a dependable team-player, mentor and friend.

Another crucial aspect of my academic career development has been the invaluable internship experiences at Toyota InfoTech Lab and Mitsubishi Electric Research Lab. I am grateful to my mentors, Dr. Ziran Wang and Dr. Marcel Menner, whose guidance has significantly broadened my knowledge in the realm of learning and control.

My graduate studies have been generously supported by the National Science Foundation. I am deeply honored to have been awarded five years of funding through the Federal Highway Administration Dwight David Eisenhower Transportation Fellowship Program. Furthermore, I am grateful to have been selected as the recipient of the Sidney P. Colowick Graduate Scholar and Harold Stirling Vanderbilt fellowships by Vanderbilt University.

Last but not the least, to my parents, grandparents, and the Stanton family – your unwavering love and support have continuously nurtured me, helping me grow into a better person. I am forever indebted to all of you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER 1

Introduction

1.1. Motivation

Complex traffic, characterized by the intricate interplay between vehicle automation technologies, human drivers, and diverse transportation modes sharing the same corridor, poses significant challenges for researchers and transportation experts. Firstly, the increasing adoption of vehicle automation technologies introduces new dynamics into traffic flow, as automated vehicles interact with human-driven vehicles. Understanding their coexistence and potential implications is essential for the safe and efficient integration of autonomous vehicles into the traffic ecosystem. Secondly, human drivers exhibit diverse driving behaviors influenced by factors such as individual personalities and decision-making processes, making it difficult to accurately capture their actions in traffic models. The complexity is further compounded by the presence of heterogeneous traffic, with vehicles of various classes and driving rules sharing the same roads. These nontrivial vehicular interactions lead to atypical flow dynamics that require advanced data analytics and modeling approaches to comprehend the intricacies of modern urban mobility. As a result, a comprehensive understanding of complex traffic necessitates the development of new estimation techniques, innovative data processing strategies, and efficient modeling approaches to capture the non-linear interactions between these elements and effectively analyze traffic behavior.

Recent advances in traffic data collection, particularly camera-based trajectory data, has revolutionized the field of understanding complex traffic systems. The vast volume and granularity of data collected from cameras and sensors provide a comprehensive view of traffic behaviors, allowing researchers to delve into microscopic details and analyze macroscopic trends. This wealth of data enables accurate calibration and validation of traffic models. Data-driven approaches, empowered by big data analytics and machine learning, unveil intricate patterns and relationships that were previously elusive. These approaches can capture the complexities of human driving behaviors, interactions between different vehicle types, and the influence of external factors like weather and road conditions. As a result, models and simulations driven by big data can predict traffic dynamics with higher precision and reliability. Real-time data processing further enhances traffic management, enabling authorities to respond dynamically to congestion and optimize traffic flow. Therefore it is important to prioritize data-processing and harness data-driven tools towards more efficient, safe, and sustainable traffic solutions.

This dissertation aims at answering the overarching question: *How can we develop modeling, estimation and data-processing methods to understand complex traffic dynamics?* In the following chapters we solve a series of reconstruction problems. Reconstruction refers to the process of recovering the unknown or missing information about a system using the available data or measurements. First we address the *parameter estimation* problem, the process of inferring the unknown parameters of a given model that describes the microscopic traffic dynamics from vehicle's on-board sensor data. Second we tackle the *state estimation* problem, which involves estimating the current or past states of macroscopic traffic dynamics based on available fixed-location measurements. Finally, we investigate *data reconciliation* problem, which processes incomplete and noisy video tracking data while ensuring compliance with dynamical constraints. The integration of these tools contributes to understanding complex traffic patterns at various scales. The specific problem formulations are

introduced next.

1.2. Problem statements

1.2.1. Driving behavior modeling

Microscopic traffic modeling and measurements that contain vehicle-level detailed dynamics are integral to the study of traffic dynamics. They reveal the relationship between individual traffic participants and the resulting traffic flow phenomena. The movement of individual vehicles based on car-following and lane-changing behavior can be characterized through microscopic models, which give a zoomed-in perspective of the traffic dynamics by looking at a small neighborhoods of vehicles. Characterizing individual vehicles can inform system-level impact such as congestion formation [13, 14, 208]. In particular, the longitudinal dynamics (or car-following that tells how each vehicle adjusts its speed in response to the change of speed and headway to the vehicle in front) provides critical safety information such as time-to-collision and safe trailing distance. Proper understanding and control of driver's longitudinal behavior has the potential to benefit traffic on a larger scale such as wave smoothing and overall safety [202, 237, 194].

The following questions remain open:

- *Can we determine the parameter identifiability of common car-following systems?* The challenges in understanding microscopic traffic dynamics stem from both the limited practical and theoretical understanding of parameter estimation performance in car-following systems. The issue of parameter identifiability, which analyzes whether the parameters can be uniquely determined given the model and data, remains insufficiently addressed. While current research mainly focuses on data-fitting quality, the fundamental question of identifiability has not been adequately explored.
- *How to develop online parameter estimation methods?* Calibrating car-following models on every vehicle-driver unit is time-consuming, and the necessary measurement data is difficult to obtain. There is a pressing need for more efficient car-following identification methods, particularly to assess the large-scale impact of mixed-autonomy traffic comprising adaptive cruise control (ACC)-enabled vehicles and human drivers. Existing techniques are predominantly offline in nature, making them less adaptable to real-time applications and lacking the flexibility to capture the nuanced driving variability. As on-board sensors increasingly provide high-frequency data and the demand for real-time driver assistance grows, there is an urgent requirement to develop online and scalable identification methods.
- *How to accurately model the intricacy of human drivers?* Accurately modeling human drivers, a major challenge, is crucial as they will remain a significant portion of road users in the near future. Human drivers exhibit inherent heterogeneity and variations influenced by various internal and external factors. Better design of vehicle automation and personalization features relies on accurate modeling of human driving behavior. The combination of physics-based and data-driven models needs to be explored to capture both variation and stochasticity of human drivers while ensuring feasibility and adherence to physical constraints.

1.2.2. **State estimation problem**

Traffic control and management strategies depend on a good estimation of traffic flow in spatial and temporal dimensions. While flow model based traffic management strategies are well developed for lane adhering homogeneous flows [57], the modelling, estimation, and control of heterogeneous traffic is less well developed. The challenges for developing traffic control and management strategies to account for increasingly heterogeneous road users are many, and must be addressed to achieve accurate, safe and effective heterogeneous traffic management. The important question becomes:

- *What can we do to accurately estimate traffic state evolution, when nontrivial vehicular interactions are involved?*

These challenges include but are not limited to (1) much more complicated vehicular interactions, such as overtaking and creeping, are difficult to capture from aggregate count or average speed data from traditional sensors; (2) lack of high quality heterogeneous traffic trajectory datasets to support research in this area. For example, although vehicle trajectory data for homogeneous traffic are abundant [159, 113, 122], they do not contain heterogeneous traffic and vehicular interactions with loose lane-discipline; (3) there is no traffic estimation routine readily available for heterogeneous or loose lane-discipline traffic, due to the increasing non-linearity of the traffic phenomena and the increased state space such that commonly used state estimators are prone to failure [24]. These limitations motivate a new generation of modelling and estimation techniques on complex traffic flows to be developed.

1.2.3. **Data reconciliation problem**

The impact of individual driving behavior (humans or automated vehicles) on broader traffic flow is relatively less studied; it has so far only been demonstrated through small-scaled experiments and simulation [194, 79, 261]. More data and representative tests can help us to come to a more realistic conclusion of their impact, and the capability to do continuous testing can inform improvements to the systems. This motivates the need for having large-scale traffic data that captures a long spatial and temporal duration.

Not only a large scope is desired, the data must also have the microscopic fidelity at the vehicle-level scale in order to validate and build realistic microscopic models. Additionally, accurate estimation of energy consumption also relies on vehicle-level detailed dynamics. However, currently available open-road trajectory datasets either lack spatial scope, or contain trajectory-level errors [123, 170, 42], which makes dynamics-based traffic analysis difficult. These challenges motivate a need for a large-scale high-quality trajectory dataset that provides a realistic picture on open-road traffic dynamics. There are issues remain to be addressed from video-based processing:

- *What are some effective approaches to handle the data quality challenges associated with existing video processing algorithms?* Current video processing algorithms suffer from inaccuracies, such as projection errors, occlusions, dimensions, and kinematic discrepancies [170]. Existing data processing techniques are ad-hoc and computationally expensive. There is a need for an efficient and systematic data processing pipeline to tackle a variety of data quality issues in video processing.
- *How to build scalable data processing pipeline to facilitate continuous data generation?* The challenge lies in creating automatic data reconciliation methods capable of handling large

volumes of trajectory data from traffic video processing. Lightweight algorithm integration is essential to ensure seamless deployment in real-world data generation systems.

1.3. **Contributions and organization of the dissertation**

Related research on addressing the above challenges are explored in Chapter 2. Chapters 3-7 document the main contributions of this thesis along with the related publications, which directly address the above challenges. Specifically:

1. Chapter 3: **Provide practical and theoretical parameter identifiability analysis of car-following systems**. This is the first work to rigorously analyze identifiability (i.e., uniqueness of parameters given input-output data) of car-following models expressed as ordinary differential equations (ODEs). Two methods are developed and applied to investigate parameter identifiability. The first method is a numerical direct test, which is a straightforward optimization problem formulation to find worst-case indistinguishable parameters in the output space. The direct test can be applied to a specific experimental setup. The second method is based on differential geometry to analyze structural identifiability of the ODEs. It gives theoretical results given generic, noise-free setting. The results indicate that all tested car-following models are structurally identifiable, but under certain experimental setup or given noisy data, they can be practically unidentifiable.

   Publications:

   - Y. Wang, M. L. Delle Monache, and D. B. Work. Identifiability of car-following dynamics. *Physica D: Nonlinear Phenomena*, 430:133090, 2022

2. Chapter 4-5: **Develop online and efficient methods for identification of ACC and human-driving car-following dynamics**. In this work an online filtering method to discover ACC car-following systems is developed, and an end-to-end data-driven model for human drivers is proposed. The proposed methods are validated with data either collected from a real vehicle or from a simulation platform, and they demonstrate more effectiveness in computation and better dynamics recovery compared to existing methods. For ACC systems, an online system identification method based on recursive least squares (RLS) is derived, which shows fast and accurate recovery of car-following dynamics. For human-driving which exhibits higher stochasticity and variance, a Gaussian Process (GP) model is learned directly from human-driving data. The model is tested in a full-scale human-in-the-loop driving simulation platform, and shows better penalization performance than existing ACCs.

   Publications:

   - Y. Wang, G. Gunter, M. Nice, M. Delle Monache, and D. Work. Online parameter estimation methods for adaptive cruise control systems. *IEEE Transactions on Intelligent Vehicles*, 6(2):288–298, 2020. URL https://doi.org/10.1109/TIV.2020.3023674

   - Y. Wang, Z. Wang, K. Han, P. Tiwari, and D. B. Work. Gaussian process-based personalized adaptive cruise control. *IEEE Transactions on Intelligent Transportation Systems*, 23(11):21178–21189, 2022

   - Y. Wang, Z. Wang, K. Han, P. Tiwari, and D. Work. Personalized adaptive cruise control via gaussian process regression. In *2021 IEEE International Intelligent Transportation*

Systems Conference (ITSC), pages 1496–1502, Indianapolis, IN, USA., 9 2021. IEEE. URL https://doi.org/10.1109/ITSC48978.2021.9564498

3. Chapter 6: **Design estimation techniques to accurately reconstruct complex, heterogeneous traffic**. We address the traffic state estimation problem in the context of heterogeneous (multi-class) traffic. The contributions made are threefold: First, we introduce a multi-class traffic estimation formulation that leverages an existing model capable of capturing intricate vehicular interactions, allowing for the simultaneous estimation of density evolution for each vehicle class. Second, we propose innovative particle filter algorithms tailored to estimate the complex evolution of mixed traffic based on sparse and noisy data, accommodating nonlinear and non-differentiable state evolution dynamics. Finally, we validate the effectiveness of these traffic estimation techniques using real traffic data collected from a mixed-traffic corridor, demonstrating their accurate recovery of traffic density evolution for each vehicle class.

   Publications:

   - Y. Wang and D. Work. Estimation for heterogeneous traffic using enhanced particle filters. *Transportmetrica A: Transport Science*, pages 1–26, 2021. URL https://doi.org/10.1080/23249935.2021.1881186

4. Chapter 7: **Propose an automatic trajectory data reconciliation pipeline for a modern traffic testbed**. We design a data reconciliation pipeline to address the limitations of video-based trajectory extraction systems. Motivated by the need for high-quality data to assess driving behavior's impact on large-scale traffic, the Tennessee Department of Transportation's I-24 MOTION testbed utilizes nearly 300 high-resolution cameras along 4.5 miles of interstate highway. The proposed pipeline automatically imputes, smooths, and corrects erroneous trajectory measurements from computer vision algorithms using an online object matching algorithm and a convex optimization formulation for individual trajectories. Once completed, this work will produce the most extensive and reliable open-road trajectory data available, benefiting mixed-autonomy traffic research and other transportation studies.

   Publications:

   - Y. Wang, D. Gloudemans, Z. N. Teoh, L. Liu, G. Zachár, W. Barbour, and D. Work. Automatic vehicle trajectory data reconstruction at scale. *arXiv preprint arXiv:2212.07907*, 2022. doi: 10.48550/ARXIV.2212.07907. URL https://arxiv.org/abs/2212.07907
   - Y. Wang, J. Ji, W. Barbour, and D. Work. Online min cost circulation for multi-object-tracking on fragments. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, 2023
   - D. Gloudemans, Y. Wang, J. Ji, G. Zachar, W. Barbour, D. B. Work, E. Hall, M. Cebelak, and L. Smith. I-24 motion: An instrument for freeway traffic science. *Transportation Research Part C: Emerging Technologies*, 2023. Under review

Lastly, Chapter 8 concludes this thesis, discusses limitations and provides future research directions.

CHAPTER 2

Related work

2.1. Microscopic traffic dynamics

2.1.1. Car-following dynamics

Microscopic-level description of traffic, such as car-following and lane-changing, provides the most detailed individual driving behavior that could have systematic impact of the overall traffic patterns. In particular, car-following dynamics describe the longitudinal dynamics of a vehicle in response to the change of speed of the vehicle in front [34, 67, 88]. Studying car-following behavior is fundamental to understand the impact on safety, fuel consumption and traffic stability [137, 136, 194, 79, 110, 146]. Physics-based control policies, expressed as ordinary differential equations (ODEs), give rise to common car-following models like the optimal velocity model (OVM) [13], intelligent driver model (IDM) [208], Gipps model [70], and Gazis-Herman-Rothery (GHR) model [34]. These ODE-based models find application in traffic microsimulation, offering provable and interpretable properties such as rational driving and stability [253, 153]. The "string stability" property of car-following systems reveals how individual driver-vehicle units impact broader traffic patterns. Identifying car-following model parameters is essential for string stability analysis and the design of string-stable ACC systems.

Parameter estimation for car-following models involves finding the best-fit parameters given input and output data from platooned vehicles, such as speed and space gaps. Various works have tackled car-following estimation for human drivers, using the NGSIM dataset [52, 172, 151]. Kesting and Treiber proposed a methodology for Intelligent Driver Model parameter estimation using a genetic search algorithm to address non-convexity in the search space [105]. Punzo and Simonelli estimated optimal parameters for different car-following models on a four-vehicle platoon using batch optimization with gradient-based techniques [169]. Besides these offline batch calibration methods, probabilistic methods offer single-pass parameter identification. Van Hinsberge et al. use Bayesian analysis to update prior probabilities into posterior probabilities [212], while Hoogendoorn & Hoogendoorn employ generalized maximum likelihood estimation [93]. Papamichail et al. summarize other microscopic model calibration approaches for automatic and connected vehicles [165]. Online parameter estimation, with potential for scalability and real-time applications, benefits from model identifiability analysis for rigorous experimental design. The convergence of car-following model parameters during estimation is influenced by the sensitivity to captured driving behavior [152]. Proper models of system and measurement noises are crucial for the performance of online filtering methods [17].

Recent learning-based approaches developed from robotics have opened up a new modeling paradigm. It motivates our study on modeling human-driving, which exhibits more variation and subtleties that the physics-based ODEs are insufficient to describe. Throughout the literature, various techniques were developed to characterize human driving styles, ranging from classification to deep learning sequence prediction. The majority treats human-driving modeling as a classification problem [234, 55, 54]. For example, drivers are categorized as normal/aggressive/cautious, based on their driving styles and skills. These class definitions are usually subjective and ambiguous, and often does not provide enough resolution to reveal their detailed car-following dynamics, which hinders

analysis such as fuel consumption, safety and driving stability. Learning-based approaches for car-following modeling have shown some new promising perspectives. A popular approach is through inverse reinforcement learning, which learns a reward function through expert demonstration [116]. Other data-driven system identification tools such as neural networks [239], SINDy [29], Gaussian Process (GP) [236] and Neuro-fuzzy methods [11] are becoming popular to identify unknown and complex systems. However, the training usually requires large amount of data, and tend to be complicated and time-consuming due to the non-parsimonious model structure, making them not suitable for time-critical applications. These tools however, are promising to be coupled with existing controllers such as Model Predictive Control (MPC) to enhance control performance and to achieve robust behaviors.

2.1.2. **Parameter identifiability**

The inverse problem of learning microscopic car following model parameters through experimental data (e.g., microscopic-level trajectory data and macroscopic level aggregated counts from fixed sensors) is part of the process to calibrate complex microscopic traffic simulation software. A recent review of these calibration techniques appears in [122], and is commonly categorized into four types [164, 171, 173, 120]. Briefly put, *type I* calibration views the calibration problem as a likelihood estimation problem where the distribution of the parameter likelihood is calculated for the future time step based on historical driving data (e.g., [164, 206, 94, 205, 173, 169, 242]). *Type II* calibration directly uses a global search to find the best-fit parameters for which the simulated complete trajectory most closely represent the observed trajectory (e.g., [133, 233, 41, 166, 81]). *Type III* calibration considers the long-range interactions amongst vehicles within a platoon (e.g., [118, 117, 85]), and *type IV* calibration relies on mesoscopic or macroscopic traffic flow patterns such as the headway distributions (e.g., [99, 120]). Chapter 3 considers parameter identifiability of microscopic models given microscopic data (e.g., velocity and spacing), which are closely related to *Type I* and *II*. More background on these *Type I* and *II* calibration methods can be found in book chapters such as [206, 37], review articles [106, 91, 94, 121, 122] and the references therein.

In addition to the question of determining the best fit parameters given experimental data, several studies have also considered the confidence level and the sensitivity of parameter estimation. For example, Punzo et al. [173] considered a variance-based global sensitivity analysis to produce the importance ranking of the IDM parameters, and to consequently reduce the parameters to be further calibrated. Monteil & Bouroche [150] considered a systematic statistical approach to first use a global sensitivity analysis to reduce the parameter space, then log likelihood estimation for the insensitive parameters and finally likelihood-ratio for interval estimation; Treiber & Kesting [205] investigated on the data sampling interval, completeness and parameter orthogonality and their effects on parameter calibration of the IDM, instead of solely on data-fitting quality. The sensitivity approaches considered in the mentioned works are closely related to identifiability [142, 150]. However, a formal analysis of identifiability on microscopic traffic models (from a theoretical perspective) has not yet been explicitly studied in the previous work.

Although a systematic study on the parameter identifiability for microscopic traffic models still remains unaddressed in transportation research, it has been developed in and extensively applied to other research fields such as waste water treatment process [21], robot dynamics [108], and biological processes [210], where the system evolution equations are also expressed as ODEs. We summarize the related approaches for tackling structural and practical identifiability respectively.

Structural identifiability of dynamical systems is closely related to the algebraic formulation of the dynamic equations, and provides theoretical possibility for uniquely inferring the system unknowns *a priori* before collecting experimental data [73, 131]. It can be analyzed using similarity transformation [227, 35, 88, 211] (applicable to autonomous systems), Laplace transform [18], power series expansion [167, 78], implicit functions [257], differential algebra [181, 58, 131] and differential geometry [220] (applicable to systems with external inputs). Multiple software packages have also been developed for direct implementation of some of the techniques, for example, DAISY [19], COMBOS [140], SIAN [92] and STRIKE-GOLDD [222, 221]. All the mentioned tools are suitable for analyzing systems of which dynamics are written as rational functions. STRIKE-GOLDD, in particular, is capable for analyzing nonlinear, non-rational system dynamics as well, making it suitable for analyzing a wider class of car-following models.

In practice when an experiment is fully defined (the specific values of the initial condition and the input trajectory are known), numerical methods are often useful to assess identifiability. The numerical methods can be used to explore the profile likelihood [177, 115], which allows to derive likelihood-based confidence intervals for each parameter and recovers the functional relations between parameters due to non-identifiability. The sensitivity matrix [195] combines numerical calculations with a tractable symbolic computation to investigate local structural identifiability.

Numerical methods can also help to assess practical identifiability, which relaxes the noise-free assumption of f and g in the structural analysis. Methods such as Monte Carlo simulations [141] help to check the relative error of estimated parameters under noisy output measurements. Numerically evaluating the profile likelihood can help to distinguish practically unidentifiable parameters due to measurement error from structurally unidentifiable ones by the shape of the likelihood profile [134].

2.2. Macroscopic traffic dynamics

2.2.1. Traffic flow models

Various macroscopic traffic flow models have been proposed to extend the seminal traffic flow models such as the *Lighthill-Whitham-Richards* (LWR) [128, 178] and *Aw, Rascle and Zhang* (ARZ) [10, 263] models, to incorporate multiple vehicle classes. One set of extensions describe multi-class traffic where vehicle classes follow homogeneous dynamics. Another set of extensions explicitly define vehicle dynamics which allow for bulk overtaking.

The growing interest for complex traffic has motivated studies on modelling traffic that is highly heterogeneous. These traffic flow models include, for example, the n-populations model [20], which assumes that the average speed of a vehicle class depends on the mean free space and allows overtaking between vehicle classes. In a related work, the porous model [154] consider the heterogeneous traffic system as porous medium which allows small and fast vehicles to move through the 'pores' defined by the free space between other vehicles in a disordered flow. Another proposed model could capture overtaking in the free flow condition [157]. The Fastlane model [214] introduces dynamic *passenger car equivalent* (PCE) parameters that scale according to the traffic state. The model of [203] considers the dynamics of traffic mixed with buses and cars. Algorithms are also developed to solve for multi-class traffic flow models, including [266, 265, 267]. Inspired by these works, the creeping model [56] explicitly defines class-specific velocity functions and jam densities to capture both overtaking and creeping features of heterogeneous traffic flows. More recently, a

coupled microscopic-macroscopic model was proposed [33] to account for the effect of large and slow-moving vehicles, while the non-local multi-class traffic flow model [39] was developed to consider heterogeneous drivers and vehicles characterised by their look-ahead visibility. The porous model [154] was re-examined [65] with an analytical expression for the pore space distribution such that the model is specifically tailored to a mixed flow of cars and powered two-wheelers.

The development of these new models are essential to incorporate the increasingly heterogeneous transportation environment present in different parts of the world. Other traffic modelling methods considering heterogeneous traffic, for example, include the multi-class model based on three dimensional flow concentration surface [148], multi-class multi-lane mesoscopic modelling [44], and cellular automata modelling [138]. More detailed reviews of these and other models can be found in [215, 62, 57].

2.2.2. **State estimation for mixed traffic**

For traffic control and management, accurate traffic state estimation is an important task. The problem is typically posed as a model-based estimation problem in which real-time data streams are used to correct model-based predictions in an online setting. *Kalman filter* (KF) and the *extended Kalman filter* (EKF) was first proposed for traffic state estimation [66, 201]. The EKF is an extension of the KF for differentiable nonlinear systems and has since been broadly applied to traffic state estimation [240, 213, 86]. For non-differentiable models such as the *cell transmission model* (CTM) or its extensions, the *unscented Kalman filter* (UKF) and the *ensemble Kalman filter* (EnKF) are also applied [86, 158, 254, 24, 179]. These Kalman-based filters, however, are minimal variance estimators which limit their application on traffic estimation problems that can generate multi-modal error distributions [24], even though error bounds can be derived [197, 223, 198]. Therefore, a fully Monte Carlo sampling-based filter, the particle filter [51, 38] is adopted [143, 144, 255, 238, 168]. Readers can refer to a complete review on traffic estimation techniques and the associated flow models in [185].

We also notice that complementary approaches on existing filters can improve the state estimation performance. Demonstrated in [143], a realistic Poisson distributed noise modeling is used to describe the empirical distribution of the field data. In [25], the randomness is introduced on the sending and receiving functions as well as on the speed adaptation rules to incorporate the stochasticity of the traffic model. Moreover, joint parameter-state estimation with a random walk parameter dynamic has shown to improve state estimation [240]; similar ideas related to dual filtering have also been explored for simultaneous parameter and state estimation [86, 213].

Unlike the widely studied homogeneous traffic flow estimation problem, only a very small number of works consider multi-class traffic state estimation, e.g., [213, 158, 156]. This is in part due to the increased complexity both in terms of the number of state variables (which increases proportionally with the number of classes), as well as the dynamics of the state variables (e.g., due to classes behaving distinctly in response to vehicles ahead). In [213], a dual EKF approach is considered to estimate the total density (and also the Fastlane model parameters), from which the individual class densities are recovered via their respective passenger car equivalents. In [156], an UKF is considered to track multi-class traffic in which overtaking is permitted in freeflow traffic but not congestion. In [158] an adaptive unscented Kalman filter that allows the model noise covariance matrix to be estimated simultaneous with the states is shown to outperform the standard UKF when applied to

freeway traffic composed of cars and trucks.

2.3. Trajectory data and historical importance

2.3.1. Vehicle trajectory datasets

Trajectories from a common reference system serve as the ultimate empirical data to study traffic dynamics. Throughout the literature, we have found a growing volume of trajectory datasets collected at various scales through different sensing mechanisms. We summarize these in Table 2.1.

Small-scale trajectory data is usually collected via vehicles that have been equipped with high-accuracy GPS sensors [97, 98, 43, 95] and other in-vehicle sensors. Unlike conventional probe vehicles or floating cars, vehicles currently collect their position data with significantly higher spatial and temporal resolution levels to obtain high-quality trajectory data [83]. Data collected from a combination of in-vehicle sensors such as radars, LiDAR and cameras provides ambient information, which can be useful for traffic flow studies as well as for training and testing AI algorithms for autonomous driving. Readers are directed to references such as [136, 102] for a comprehensive discussion on such dataset.

Image sensors and video processing algorithms associated with overhead cameras or Unmanned Aerial Systems (UAV) have become increasingly powerful as a large-scale trajectory data source. The seminal work is the Next Generation Simulation (NGSIM) dataset [159] developed in early 2000s, which is a collection of real-world trajectories, based on the use of cameras mounted on tall buildings and covering approximately 600-meter long roadway sections with a frequency of 10 Hz in several US locations. More recently in the highD dataset [114], videos were recorded by overhead flying drones (4096×2160 pixels, 25 fps), which provided much higher-quality videos than that of the NGSIM dataset [42]. HighD provides highway trajectories covering 420m of road segment, for a total of 28,000 vehicle miles traveled. The pNEUMA large-scale field experiment [16] recorded traffic streams in a multi-modal congested environment over an urban area using UAV. The dataset was generated by a swarm of 10 drones hovering over a traffic intensive area of 1.3 km^2 in the city center of Athens, Greece, covering more than 100 km-lanes of road network at 25Hz. Most recently, Automatum dataset became available [190], which covers 12 characteristic highway-like scenes from 30 hours of drone videos. HIGH-SIM [186] is another high-quality highway trajectory dataset extracted from aerial video data. The videos were collected by three 8K cameras in a helicopter over an 8000 ft long segment of the I-75 freeway in Florida for 2 hours. The dataset covers a wide range of traffic characteristics. A comparison across selected datasets can be viewed from Table 2.1. Other examples and discussion on their usage in traffic studies can be found in a comprehensive survey, such as [123].

2.3.2. Data quality issues and remedies

Raw trajectories from video footage are usually obtained from a *tracking-by-detection* framework [100, 7], where a set of object detections for all frames are linked across time to form continuous trajectories. The detected bounding boxes are then transformed to a common reference system (e.g., global positioning system) for meaningful interpretation of dynamics. One of the errors we aim to reconcile is tracking discontinuity due to, for example, noisy detection, inevitable object occlusion and switching camera field of views, which results in track fragmentations. The first problem to

Dataset	Year	Context	Camera config	Road segment	VMT	Hours of recording
NGSIM [159]	2006	Highway	8 cameras	600m	18,000	0.75
pNeuma [16]	2018	Urban arterial	10 drones	10km	N/A	59
highD [114]	2018	Highway	1 drone	420m	28,000	147
Automatum [190]	2021	Highway	Overhead drones	N/A	18,724	30
HIGH-SIM [186]	2021	Highway	Aerial cameras	2,438m	N/A	2
I-24 MOTION [75]	2022	Highway	276 4K cameras	4.2 miles	200M/yr (expected)	daylight

Table 2.1: Video-based trajectory data comparison. VMT: vehicle miles traveled

address is data association, with is given a set of detections, identify their origin (objects that they are associated with). Data association problems can be difficult when the tracking produces large fragmentations or when the detections have false alarms or missing detections, which are common in real-world video-based detection systems.

Various approaches for data association have been proposed, taking into consideration factors such as association criteria, object motion complexity, and computational requirements. These approaches typically differ in their choices of 1) matching cost (referred to as probability, affinity, energy, or confidence) and 2) matching criteria (such as global cost minimization, greedy approach, hierarchical matching, etc.), leading to different problem formulations. The matching cost incorporates kinematic information (e.g., position and velocity) and attribute information (e.g., shape and appearance), while the matching criteria guide the algorithm in solving the data association problem.

Solving the optimal data association or multi-object-tracking (MOT) problem is inherently a challenging NP-hard matching problem that requires combinatorial optimization algorithms. However, specific characteristics of the MOT problem, such as the Markov assumption of association cost, can be leveraged to apply polynomial-time algorithms like bipartite matching and min-cost flow solvers. Graph-based formulations offer efficient algorithms for finding global minimum-cost tracking solutions. In these formulations, tracks (or detections) are represented as nodes in a graph, while pairwise matching costs are represented as graph edges. The general data association problem can be viewed as finding the least-cost set cover on the track graph [40]. Several studies [32, 264, 224] have investigated efficient algorithms related to bipartite matching and min-cost flow. Interested readers are encouraged to refer to a recent survey [175] for further exploration of this topic.

To the best of our knowledge, the majority of previous studies on graph-based MOT approaches have focused on offline methods. In these methods, all detections/fragments must be available in memory to construct a static graph. However, this approach can be a significant disadvantage, especially as more sensing devices provide sequential data that requires continuous monitoring. While a few online methods, such as [119], operate on a frame-by-frame basis, they still require multiple iterations of updates within each frame. Therefore, it is necessary to develop an online method that operates on a fragments (or tracklets) graph. This means that fragments are added to the graph one at a time, resulting in even further reduced computation requirements.

Although trajectory data serves as the fundamental building block for traffic studies, many works [42, 170, 204] raised the issue of the quality of the seminal NGSIM data, including nonphysical kinematics and inter-vehicle distance. The importance for data postprocessing is also emphasized in [106, 170, 207, 123]. Montanino and Punzo [149] undertook one of the most thorough efforts of data reconciliation by applying a series of smoothing operations on the dataset. They posed the smoothing and reconstruction as a nonlinear, non-convex optimization problem, as to find the minimum local smoothing window subject to kinematic constraints. Other data postprocessing efforts [204, 53] also performed data reconciliation considering realistic vehicle kinematics and driving dynamics. Although comprehensive, these works provide ad-hoc treatments for specific sources of errors and rely on multiple iterations of smoothing, which cannot be applied for the streaming setting. We are in need of a more efficient alternative for trajectory reconciliation that applies to streaming data.

CHAPTER 3

Microscopic dynamics reconstruction and parameter identifiability

3.1. Introduction

With the advancement of sensor technologies, abundant traffic data is readily available to study traffic patterns and individual driving behaviors. Examples of such datasets include overhead camera data [159, 113] and floating-car data [169, 206, 2, 30]. The collection of such datasets support tasks such as the parameterization of microscopic models which describe individual vehicle car-following behaviors.

Studies of car-following models and calibration of such models have mostly been focused on data-fitting quality. Model calibration is usually posed as an optimization problem such that the best fit parameters are found by minimizing the error between the model prediction and corresponding measurement data [107, 169], or through probabilistic approaches to find the most likely parameter candidate [242, 1]. Although the approaches report good accuracy of the estimated parameters, they lack a theoretical guarantee that a unique parameter set can be recovered, which is provided by an identifiability analysis of the evolution-observation system [18, 130]. In other words, using data-fitting quality as the metric alone for model calibration does not guarantee the uniqueness of the best-fit parameters, nor can it tell how robust the current estimation is under a different experimental dataset or numerical setup [130]. Such ambiguity might result in a different parametrization when repeating the experiment, leading to an inaccurate characterization of car-following behaviors, such as analytically determining string stability [153].

Besides that different driver behaviors on a microscopic scale can greatly change traffic features on a macroscopic scale [118, 120, 169], knowledge of microscopic model parameters is key to understand individual driving behavior of the adaptive cruise control and the interplay between automated vehicles and human drivers [13, 67, 81, 146, 147, 242, 85, 107]. Therefore, the identifiability of car-following models is an important question to advance calibration of these models.

3.2. Problem statement

The parameter identifiability problem refers to the ability to uniquely recover the unknown model parameters from the observation data. Mathematically, consider a general form of a dynamical system:

$$\begin{cases} \dot{x}(t) = f(x(t), u(t), \theta) \\ y(t) = g(x(t), u(t), \theta) \\ x(0) = x_0, \end{cases} \tag{3.1}$$

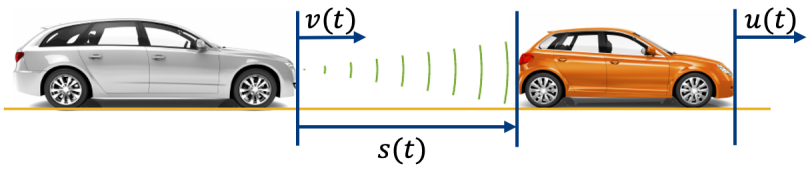


Figure 3.1: A car-following system.

where $x(t) \in \mathbb{R}^{n_x}$ is the state variable vector, $u(t)$ is the scalar input, and $\theta \in \mathbb{R}^{n_\theta}$ is the parameter vector; f, g are analytic vector functions describing the evolution of the state and the measurement model, and x_0 is the initial condition. The generic question for parameter identifiability is: *Given the system* (3.1) *and a known input* $u(t)$*, can we uniquely determine the model parameters* θ *from the output* $y(t)$ *for a given initial condition?* In the context of car-following (Figure 3.1) models, the system of equations can be written as:

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} \dot{s}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} u(t) - v(t) \\ f_{\mathrm{CF}}(x(t), u(t), \theta) \end{bmatrix} \\ y(t) = x(t) \\ x(0) = \begin{bmatrix} s_0 \\ v_0 \end{bmatrix}. \end{cases} \tag{3.2}$$

The state vector $x(t)$ in (3.2) is given by the space gap $s(t)$ and the velocity $v(t)$ of the ego vehicle. The space gap is taken as the distance between the lead vehicle and the ego (follower) vehicle while the speed is that of the ego vehicle. The input $u(t)$ is the velocity profile of a lead vehicle and f_{CF} is usually a 2^{nd} order ordinary differential equation (ODE) that describes the car-following dynamics.

With in-vehicle sensor data such as radar data being available, it is now possible to directly measure $s(t)$ and $v(t)$, as well as the velocity profile of the lead vehicle $u(t)$. For (3.2), the parameter identifiability question becomes: *Given the output time-series* $y(t)$ *(the velocity and space gap measurements) and the input* $u(t)$*, can we uniquely determine the model parameters* θ *in* f_{CF}*?*

One type of identifiability is *structural identifiability* (or theoretical identifiability), which investigates the input-output configuration and relies on algebraic calculations to determine if the parameter set is unique from the observation [229, 131]. It is completely defined by the model structure and does not consider data quality or model errors. A *structural identifiability* analysis can be performed prior to the estimation of the unknowns in order to detect potential model structural issues and separate them from numerical problems, which should be dealt with differently [222]. However, structurally identifiable systems do not preclude the existence of some specific initial conditions under which the system becomes unidentifiable [183, 218]. For this limitation, *practical identifiability* can be complimentary to provide insights on the dynamical models.

Practical identifiability is usually based on numerical methods to determine, for a given experimental setup, whether the parameters are unique. This is sometimes modified to also consider the presence of moderate measurement errors. In practice, a realistic way to formulate the identifiability problem when performing an experiment becomes: *for a given initial condition and an input trajectory that are known, whether there exists distinct valid parameters that produce the same outputs?* Notice that ensuring uniqueness of the parameters (e.g. there are no distinct parameters that produce the same output) does not imply well-posedness of the inverse problem. When solving the inverse problem, one needs to ensure additional properties, e.g. continuity w.r.t. the initial datum, to guarantee the recovery of the parameters and reconstruction the input.

This chapter investigates, from both theoretical and practical angles, parameter identifiability of a microscopic car-following system instantiated as one of four different car-following models, i.e., the constant-time headway relative-velocity (CTH-RV) model [13, 146, 81], the optimal velocity (OV) model [13], the follow-the-leader (FTL) model [63] and the intelligent driver model (IDM)[208]. Only CTH-RV is a linear model while other three are nonlinear. We address the following questions:

- Are the car-following systems of the form (3.2) structurally identifiable?
- Are they identifiable in practice given a specific experimental setup (i.e., given a known initial condition and a known input)?
- Are they identifiable in practice given a specific experimental setup when moderate measurement errors are allowed?

We address the question of structural identifiability using a *differential geometry approach*, and practical identifiability using a *numerical direct test*. The main findings to the above three questions are:

- The investigated car-following models are all structurally locally identifiable, i.e., almost all points in the parameter space of each of the car following models can be uniquely identified for all admissible inputs given almost all initial conditions. Special initial conditions and parameter sets that lose structural identifiability are also discovered.
- Given a specific experimental setup (the specific initial condition and input that are known), the CTH-RV and FTL are not identifiable, i.e., there exist distinct parameters that produce the same output.
- Given a specific experimental setup when measurement errors are present, all models are not practically identifiable, i.e., there exist distinct parameters that produce the same output within a small error bound.

Throughout the literature of microscopic traffic modeling we found that the focus has been on model calibration techniques instead of parameter identifiability. In this light, the main contributions of this chapter are the following:

- We provide structural identifiability analysis of four common car-following models using a differential geometry framework. This allows us to detect if there are structural issues of the dynamic model and to distinguish them from other possible causes of calibration failures, such as the choice of optimization algorithms.
- We subsequently provide a numerical *direct test*, which is a constrained optimization problem, to check identifiability for a specific experimental design (for a given and known initial condition and input.
- We investigate the practical identifiability for a specific experimental design when moderate measurement errors are present, also with a *direct test*.

This study therefore provides both analytical and practical insights on parameter identification.

The organization of this chapter is as follows. In Section 2.1.2 we summarize the existing microscopic traffic model calibration work, as well as methods derived in other research fields to tackle parameter identifiability problems. In Section 3.3 we present the four microscopic car-following models that we analyze in this work. We introduce the methodological tools used, including the *differential geometry* approach and the *numerical direct test* approach, in Section 3.4. Section 3.5 provides the structural identifiability results of the selected car-following models, and Section 3.6 presents the practical identifiability results.

3.3. **Car-following models**

In this section, we briefly present four commonly used car-following models that describe the acceleration dynamics in the form of an ODE.

Constant-time headway relative-velocity (CTH-RV) model The CTH-RV model has been predominately used to describe the car-following behaviors of ACC vehicles [146, 147, 81, 242, 15, 127]. It is a simple model with linear dynamics with respect to the space gap and relative velocity $\Delta v(t) := u(t) - v(t)$:

$$\dot{v}(t) = k_1(s(t) - \tau v(t)) + k_2(\Delta v(t)), \tag{3.3}$$

where k_1 and k_2 are non-negative gains and τ is the constant time-headway. The three (time-invariant) parameters constitute the parameter vector of this model, i.e., $\theta = [k_1, k_2, \tau]$, whose identifiability is to be determined.

The equilibrium initial condition x_0^* for the car-following system (3.2) under the acceleration dynamic (3.3) is:

$$x_0^* = [\tau u_0, u_0]^T, \tag{3.4}$$

where $u(0) = u_0$ is the initial velocity of the leader. It is easy to see that this initial condition results in $\dot{s}(0) = u_0 - v_0 = 0$ and $\dot{v}(0) = f_{CF}(x_0^*, u_0, \theta) = 0$.

Optimal velocity (OV) model We also consider the car-following model proposed by Bando et al. [12, 13], where an optimal velocity term $V(s(t))$ is introduced to describe the desired spacing-speed relationship at equilibrium:

$$V(s(t)) = a\left(\tanh\left(\frac{s(t) - h_m}{b}\right) + \tanh\left(\frac{h_m}{b}\right)\right), \tag{3.5}$$

where the parameters a, h_m, and b determine the shape of the optimal velocity function, which increases monotonically as a function of $s(t)$, and asymptotically plateaus at a maximum speed as $s \to \infty$. Consequently the vehicle accelerates and decelerates to achieve the optimal velocity:

$$\dot{v}(t) = \alpha\left(V(s(t)) - v(t)\right), \tag{3.6}$$

where the parameter α determines the sensitivity of the stimulus, which is the difference between the desired velocity $V(s(t))$ and the actual velocity $v(t)$. The parameter vector of this model is $\theta = [\alpha, a, h_m, b]$. The corresponding equilibrium initial condition for (3.2) under (3.6) is:

$$x_0^* = \left[h_m - b * \tanh^{-1}\left(\tanh\left(\frac{h_m}{b}\right) - \frac{u_0}{a}\right), u_0\right]^T. \tag{3.7}$$

Follow-the-Leader (FTL) model The third model is one of the simplest follow-the-leader variations of the Gazis-Herman-Rothery (GHR) car-following model, which originated from research conducted by General Motors in the 1950s [67, 69, 63]. We consider the following form:

$$\dot{v}(t) = \frac{C\Delta v(t)}{s(t)^\gamma}, \tag{3.8}$$

where the parameter C and γ are constants describing the sensitivity of $\Delta v(t)$ and $s(t)$, respectively, or the acceleration. The parameter vector of interest is $\theta = [C, \gamma]$. The corresponding equilibrium initial condition x_0^* for (3.2) under (3.8) is:

$$x_0^* = [s_0, u_0]^T. \tag{3.9}$$

Intelligent driver model (IDM) The intelligent driver model was proposed in [208] to model a realistic driver behavior, such as asymmetric accelerations and decelerations. It is of the form:

$$\dot{v}(t) = a\left[1 - \left(\frac{v(t)}{v_f}\right)^4 - \left(\frac{s^*(v(t), \Delta v(t))}{s(t)}\right)^2\right] \tag{3.10}$$

where the desired space gap s^* is defined as:

$$s^*(v(t), \Delta v(t)) = s_j + v(t)T + \frac{v(t)\Delta v}{2\sqrt{ab}}. \tag{3.11}$$

The parameters of the model $\theta = [s_j, v_f, T, a, b]$ are the freeflow speed v_f, the desired time gap T, the jam distance s_j, the maximum acceleration a and the desired deceleration b. The corresponding equilibrium initial condition x_0^* is:

$$x_0^* = \left[\frac{s_j + u_0 * T}{\sqrt{1 - (u_0/v_f)^4}}, u_0\right]^T. \tag{3.12}$$

3.4. Methodology

In this section we highlight the methods for identifiability analysis. The structural local identifiability is carried out through a *differential geometry* method, and the practical identifiability considering a full experimental setup and measurement error is examined using a *numerical direct test* method.

3.4.1. Differential geometry framework for structural identifiability

We first introduce the concepts of structurally global and structurally local identifiability via precise definitions. Note that the following definitions are modified from references such as [222, 78, 131, 228], in order to suit the specific format of our analysis.

Definition 1. Let $\theta \in \mathbb{R}^{n_p}$ denote the generic parameter vector, $\mathcal{X}_0$ a set of generic initial conditions and $\mathcal{U}$ a set of admissible inputs. Let $y(t, \theta, x_0, u)$ be the output function from the state-space model (3.1). If for all $t > 0$,

$$y(t, \theta, x_0, u) \equiv y(t, \theta^*, x_0, u) \Rightarrow \theta = \theta^* \tag{3.13}$$

for almost all $\theta \in \mathbb{R}^{n_p}$, almost all $x_0 \in \mathcal{X}_0$ and every input $u \in \mathcal{U}$ then the model is said to be structurally globally identifiable.

Definition 2. A dynamical system given by (1) is structurally locally identifiable (s.l.i.) if for almost all $\theta \in \mathbb{R}^{n_p}$ there exists a neighborhood $\mathcal{N}(\theta)$ such that, for all $\theta_1, \theta_2 \in \mathcal{N}(\theta)$, the implication (3.13) holds for all $t > 0$.

Remark 1. Notice that Definitions 1. and 2. consider the generic parameter vector and initial conditions, and thus may not be applicable to parameters and initial conditions that fall into a measure zero set.

Remark 2. We consider admissible inputs are polynomial inputs of degree n (with n non-zero time derivative) that enables structurally identifiable systems. In particular, no admissible input exists for structurally unidentifiable systems.

The differential geometry framework [217] considers identifiability as an augmented observability property for a general nonlinear system of ODEs such as (3.1), and it can be evaluated in the same manner. The main idea of identifiability consists in considering the parameters θ as additional states with zero dynamics. Hence, the parameter-augmented state and the dynamics become $\tilde{x} = [x, \theta] \in \mathbb{R}^{n_{\tilde{x}}}$, with $n_{\tilde{x}} = n_x + n_\theta$ and

$$\tilde{x}(t) = \begin{bmatrix} x(t) \\ \theta \end{bmatrix} \qquad \text{and} \qquad \begin{cases} \dot{\tilde{x}}(t) = f(\tilde{x}(t), u(t)) \\ y(t) = g(\tilde{x}(t), u(t)) \\ \tilde{x}(0) = \tilde{x}_0 = [x_0, \theta]^T. \end{cases} \tag{3.14}$$

In the general case of a nonlinear system with time-varying input, to identify the parameters we need to take into account the changes to the output due to the changing input in the augmented state. Thus we use the extended Lie derivatives. The extended Lie derivatives can, then, be used to build the rows of the observability-identifiability matrix $\mathcal{O}_I$ [103, 221], whose rank is to be evaluated for the identifiability analysis. The Lie derivative of $g(\tilde{x}(t), u(t))$ in the direction of $f(\tilde{x}(t), u(t))$ is given by:

$$L_f g(\tilde{x}, u) = \frac{\partial g(\tilde{x}, u)}{\partial \tilde{x}} f(\tilde{x}, u) + \sum_{j=0}^{j=\infty} \frac{\partial g(\tilde{x}, u)}{\partial u^{(j)}} u^{(j+1)} \tag{3.15}$$

where $u^{(j)}$ is the j^{th} time derivative of the input u. The Lie derivatives of higher order are:

$$L_f^i g(\tilde{x}, u) = \frac{\partial L_f^{i-1} g(\tilde{x}, u)}{\partial \tilde{x}} f(\tilde{x}, u) + \sum_{j=0}^{j=\infty} \frac{\partial L_f^{i-1} g(\tilde{x}, u)}{\partial u^{(j)}} u^{(j+1)}. \tag{3.16}$$

The observability-identifiability matrix $\mathcal{O}_I(\tilde{x}, u)$ for a general nonlinear system (3.14) becomes

$$\mathcal{O}_I(\tilde{x}, u) = \begin{pmatrix} \frac{\partial}{\partial \tilde{x}} g(\tilde{x}, u) \\ \frac{\partial}{\partial \tilde{x}} (L_f g(\tilde{x}, u)) \\ \frac{\partial}{\partial \tilde{x}} (L_f^2 g(\tilde{x}, u)) \\ \vdots \\ \frac{\partial}{\partial \tilde{x}} (L_f^{n_{\tilde{x}}-1} g(\tilde{x}, u)) \end{pmatrix} \tag{3.17}$$

Finally, a nonlinear observability-identifiability condition can be used to infer the structural local identifiability of (3.14):

Theorem 1 (Nonlinear Observability – Identifiability Condition (OIC))**.** If a model (3.14) satisfies $\text{rank}(\mathcal{O}_I(\tilde{x}_0, u)) = n_x + n_\theta$, with $\mathcal{O}_I(\tilde{x}_0, u)$ given by (3.17) and $\tilde{x}_0$ being a (possibly generic) point in the augmented state space, the model is locally observable and locally structurally identifiable in a neighborhood $\mathcal{N}(\tilde{x}_0)$ of $\tilde{x}_0$.

Remark 3. The differential geometry approach yields results that are valid almost everywhere, i.e., for all values of the system variables (initial conditions and inputs) except for a set of measure zero.

Throughout the rest of the chapter, we focus on analyzing structural local identifiability for almost all initial conditions in the augmented initial state $\tilde{x}_0$, i.e., we compute $\mathcal{O}_I(\tilde{x}_0, u)$ symbolically with $\tilde{x}_0 = [x_0, \theta]^T$. We also explore the special cases for $\tilde{x}_0$ (the initial conditions and parameters that belong to a set of measure zero) such that structural identifiability is lost. More details of the setup and analysis are presented in Section 3.5.

3.4.2. **Numerical direct test**

The *direct test* is a conceptually straightforward way to test identifiability considering a fully-defined system (i.e., the given initial condition and input trajectory are known). The direct test offers analysis that differs from structural identifiability in that it looks for the worst-case unidentifiable parameters in the following sense. It poses the identifiability problem as for finding the maximally distinct (as specified by an objective function) parameters θ_1 and θ_2 such that the output of two systems (which are identical except for the parameters) differ by no more than a threshold ϵ (possibly equal to zero). It is an idea first proposed in [230] as a numerical alternative to algebraic computation.

The generic form of the direct test reads:

$$\begin{aligned} \underset{\theta_1,\theta_2\in\Theta}{\text{maximize}} \quad & d(\theta_1, \theta_2) \\ \text{subject to} \quad & e(\theta_1, \theta_2) \leq \epsilon, \end{aligned} \tag{3.18}$$

where the objective function $d(\theta_1, \theta_2)$ is the distance between two parameters in the parameter space Θ. The constraint $e(\theta_1, \theta_2) \leq \epsilon$ caps the difference between the output of the system under θ_1 and θ_2 at ϵ, which is a small, user-defined threshold of measurement error. In order to evaluate $e(\theta_1, \theta_2)$, one must solve the ODE (3.1) for the same known initial condition and the same given input trajectory under the two parameters θ_1 and θ_2. This problem formulation finds the most distinct (as quantified by d) parameters such that the fully defined system produces similar outputs (or the same outputs when $\epsilon = 0$). The difference $e(\cdot, \cdot)$ is only due to θ for a fully-defined system. Therefore $e(\cdot, \cdot)$ is a function of only two parameters θ_1 and θ_2.

Denote the decision variables of (3.18) at optimality as θ_1^*, θ_2^*, and the objective function value at optimality as $\delta^* := d(\theta_1^*, \theta_2^*)$. Naturally, under the tightest constraint $\epsilon = 0$, all parameters in Θ for system (1) with a specific and known initial condition and input trajectory produce a unique output if and only if $\delta^* = 0$. When this occurs, we say that the system (3.1) with the prescribed initial and input condition is identifiable; the same system is unidentifiable under the same initial and input conditions if $\delta^* > \delta$, where δ is a small positive number that is problem-specific.

When measurement error is allowed, we consider a relaxed constraint, $\epsilon > 0$. If δ^* is small, then we say the system is practically identifiable, i.e., similar outputs can be produced only by parameters that are similar. On the other hand, if δ^* is large, we found two distinct parameters that generate similar outputs, indicating that the same system is practically unidentifiable.

Results on the change of δ^* as ϵ changes provides insights on the sensitivity of identifiability with respect to the measurement error, and will be presented in the numerical experiment (Section 3.6).

3.5. **Structural identifiability analysis**

In this section, we analyze structural local identifiability of the car-following system (3.2) under the various car following models in Section 3.3. The analysis reveals that all the systems are structurally locally identifiable, i.e., it is theoretically possible to uniquely infer the unknown parameters under any admissible input and almost all initial conditions. We also explore the specific initial conditions (those belonging to a set of measure zero) such that each model becomes unidentifiable.

The organization of this section is the following. First we give an overview of the analysis, including the toolbox for implementing the differential geometry analysis for a given car following model, and interpret the results of the analysis. Next, we provide the structural local identifiability analysis for the CTH-RV (3.3) model in detail, by showing the explicit observability-identifiability matrices $\mathcal{O}_I(\tilde{x}_0, u)$ under both a generic and specific initial conditions, and a set of admissible inputs for which the system is structurally identifiable. The structural local identifiability results of the other models ((3.6), (3.8) and (3.10)) are then summarized.

3.5.1. **Implementation overview**

We deploy a direct implementation of the framework for structural identifiability through *STRuctural Identifiability taKen as Extended-Generalized Observability with Lie Derivatives and Decomposition* (STRIKE-GOLDD) [219, 222], an open-source MATLAB toolbox that computes the observability-identifiability matrix $\mathcal{O}_I(\tilde{x}_0, u)$ and analyzes the local structural parameter identifiability, state observability, and input reconstructability of nonlinear dynamic models of ODEs. Because only parameter identifiability is concerned in this chapter as the states and the inputs can be directly measured, we use the toolbox solely for the purpose of identifiability.

In addition, we use STRIKE-GOLDD to also determine *a priori* the minimum degree n of a polynomial input $u(t)$ to enable a structurally identifiable system [221]. For example, if $\mathcal{O}_I$ has full rank under $u^{(n)}(t) = 0$ but not $u^{(n-1)}(t) = 0$, then an input $u(t)$ of degree n or above is sufficient for structural identifiability. Otherwise, $u(t)$ with degree higher than n should be explored.

For structurally unidentifiable systems, no admissible input exists. For structurally identifiable systems, the degree n (i.e., number of non-zero time derivatives) for an input to be admissible depends on the initial condition. We explore the admissible input condition for structurally identifiable car-following systems under both generic initial conditions and a special set of initial conditions where higher-order input may be required to be admissible.

3.5.2. **Structural local identifiability analysis for CTH-RV**

Recall CTH-RV ODE (3.3). The parameter-augmented system dynamics can be written as

$$\begin{cases} \widetilde{\dot{x}}_1(t) = \dot{s}(t) = u(t) - v(t) \\ \widetilde{\dot{x}}_2(t) = \dot{v}(t) = k_1(s(t) - \tau v(t)) + k_2(u(t) - v(t)) \\ \widetilde{\dot{x}}_{3-5}(t) = \dot{\theta}(t) = 0 \\ y(t) = s(t) \\ \tilde{x}(0) = [x_0, \theta]^T. \end{cases} \tag{3.19}$$

The following propositions and proofs show the structural identifiability of CTH-RV from the calculated observability-identifiability matrices.

Proposition 1. According to Definition 2, the CTH-RV system (3.19) is structurally locally identifiable under almost all initial condition x_0 and an admissible input $u(t)$ with degree $n \geq 0$ up to a set of measure zero.

Proof. Consider almost all initial state conditions $x_0 = [s_0, v_0]^T$. We prove that input $u(t)$ with degree $n \geq 0$ is admissible for structural local identifiability, i.e., $\mathcal{O}_I(\tilde{x}_0, u)$ is full rank with constant input $u(t) = u_0$. This implies that $\mathcal{O}_I(\tilde{x}_0, u)$ for $n > 0$ is also full rank [221].

Constructing the observability-identifiability matrix $\mathcal{O}_I(\tilde{x}_0, u)$ requires at least $n_{\tilde{x}} - 1 = 4$ extended Lie Derivatives, where $n_{\tilde{x}} = 5$ is the dimension of the augmented state. The four extended Lie Derivatives are:

$$\begin{aligned}
L_f^1 g(\tilde{x}, u) &= u(t) - v(t) \\
L_f^2 g(\tilde{x}, u) &= k_2(v(t) - u(t)) - k_1(s(t) - \tau v(t)) \\
L_f^3 g(\tilde{x}, u) &= k_1(v(t) - u(t)) - (k_2 + k_1\tau)(k_2(v(t) - u(t)) - k_1(s(t) - \tau v(t))) \\
L_f^4 g(\tilde{x}, u) &= -(k_1 - (k_2 + k_1\tau)^2)(k_2 * (v(t) - u(t)) - k_1(s(t) - \tau v(t))) \\
&\quad - k_1(k_2 + k_1\tau)(v(t) - u(t)).
\end{aligned} \tag{3.20}$$

Consequently, $\mathcal{O}_I(\tilde{x}_0, u)$ for system (3.19) is a 5×5 matrix:

$$\mathcal{O}_I(\tilde{x}_0, u) = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & 0 & 0 \\
-k_1 & k_2 + k_1\tau & \tau v_0 - s_0 & v_0 - u_0 & k_1 v_0 \\
-k_1(k_2 + k_1\tau) & (k_2 + k_1\tau)^2 - k_1 & o_{43} & o_{44} & o_{45} \\
o_{51} & o_{52} & o_{53} & o_{54} & o_{55}
\end{bmatrix} \tag{3.21}$$

with specific entries defined as follows:

$$\begin{aligned}
o_{43} &= v_0 - u_0 + (k_2 + k_1\tau)(s_0 - \tau v_0) - \tau(k_2(v_0 - u_0)) - k_1(s_0 - \tau v_0) \\
o_{44} &= k_1(s_0 - \tau v_0) - k_2(v_0 - u_0) - (k_2 + k_1\tau)(v_0 - u_0) \\
o_{45} &= -k_1(k_2(v_0 - u_0) - k_1(s_0 - \tau v_0)) - k_1 v_0(k_2 + k_1\tau) \\
o_{51} &= k_1(k_1 - (k_2 + k_1\tau)^2) \\
o_{52} &= -k_1(k_2 + k_1\tau) - (k_2 + k_1\tau)(k_1 - (k_2 + k_1\tau)^2) \\
o_{53} &= (s - \tau v_0)(k_1 - (k_2 + k_1\tau)^2) - (k_2 + k_1\tau)(v_0 - u_0) + (2\tau(k_2 + k_1\tau) - 1) \\
&\quad (k_2(v_0 - u_0) - k_1(s_0 - \tau v_0)) - k_1\tau(v_0 - u_0) \\
o_{54} &= (2k_2 + 2k_1\tau)(k_2(v_0 - u_0) - k_1(s_0 - \tau v_0)) - \\
&\quad k_1(v_0 - u_0) - (k_1 - (k_2 + k_1\tau)^2)(v_0 - u_0) \\
o_{55} &= k_1^2(v_0 - u_0) - k_1 v_0(k_1 - (k_2 + k_1\tau)^2).
\end{aligned}$$

Analytically, it is difficult to prove that the matrix $\mathcal{O}_I(\tilde{x}_0, u)$ is full rank. A symbolic calculator such as MATLAB can provide a general case: the matrix in general is not rank deficient (see an example in Appendix A). In this case, the system is structurally locally identifiable according to Theorem 1. Since constant input $u(t) = u_0$ is admissible, $u(t)$ with degree $n \geq 0$ is also admissible according to [221]. □

However, Theorem 1 provides a general result that is valid for all values of the state and parameters except for a set of measure zero. It is possible that for some special initial conditions the matrix is not full rank. The symbolic calculator leads to generic conclusions on the rank of a matrix and fails to capture a specific set (of measure zero). Therefore, it is possible to categorize the system as locally structurally identifiable whereas for special initial conditions the result is uninformative. The following proposition gives an example of special initial condition that leads to unidentifiable CTH-RV system:

Proposition 2. The CTH-RV system (3.19) is unidentifiable given an equilibrium initial condition (3.22) and constant input $u(t) = u_0$.

Proof. We prove that the system (3.19) is unidentifiable under constant input $u(t) = u_0$ with an equilibrium initial conditions specified as

$$x(0) = x_0^* = \begin{bmatrix} \tau u_0 \\ u_0 \end{bmatrix}. \tag{3.22}$$

where u_0 is the initial value of the lead vehicle velocity. The equilibrium initial condition results in $\dot{s}(0) = 0$ and $\dot{v}(0) = f_{\text{CTHRV}}(u_0, x_0^*, \theta) = 0$. The same four Lie Derivatives (3.20) will be used to calculate $\mathcal{O}_I(\tilde{x}_0, u)$. However, due to the specified initial conditions and the constant input condition, $\mathcal{O}_I(\tilde{x}_0, u)$ becomes:

$$\mathcal{O}_I(\tilde{x}_0, u) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -k_1 & k_2 + k_1\tau & 0 & 0 & k_1 u_0 \\ k_1(k_2 + k_1\tau) & k_1 - (k_2 + k_1\tau)^2 & 0 & 0 & -k_1 u_0 (k_2 + k_1\tau) \\ o_{51} & o_{52} & 0 & 0 & o_{55} \end{bmatrix} \tag{3.23}$$

$$\begin{aligned} o_{51} &= k_1(k_1 - (k_2 + k_1\tau)^2) \\ o_{52} &= -k_1(k_2 + k_1\tau) - (k_2 + k_1\tau)(k_1 - (k_2 + k_1\tau)^2) \\ o_{55} &= -k_1 u_0 (k_1 - (k_2 + k_1\tau)^2). \end{aligned}$$

It is obvious that $\text{rank}(\mathcal{O}_I(\tilde{x}_0, u)) = 3$, and therefore this system is structurally unidentifiable under equilibrium initial conditions. Furthermore, the specific unidentifiable parameter(s) can be detected by removing each of the columns of $\mathcal{O}_I(\tilde{x}_0, u)$ and recalculating the rank. If the rank does not change after removing the ith column, than the corresponding ith variable in the augmented state $\tilde{x}$ is unidentifiable (or observable) [222]. It is straightforward to see that removing the 3rd or the 4th column of $\mathcal{O}_I(\tilde{x}_0, u)$ does not change the rank, and therefore, the corresponding parameters k_1 and k_2 are unidentifiable. □

Corollary 1. Given an equilibrium initial condition x_0^* for the CTH-RV system (3.19), the admissible input required to enable structural identifiability is with degree $n \geq 1$.

In the case of equilibrium initial condition, a constant input is not admissible to enable structural identifiability. Therefore, time-varying input may be needed to excite the system. We check the

order of the input required by replacing the higher order derivatives of $u(t)$ in $\mathcal{O}_I$ with zero and recalculating rank($\mathcal{O}_I$). This procedure gives a full rank ($\mathcal{O}_I$) when $\ddot{u} = 0$, as long as $\dot{u} \neq 0$. Therefore an input with degree $n \geq 1$ is admissible for structural identifiability of CTH-RV, even if the system starts from an equilibrium initial condition.

Another unidentifiable case that *differential geometry* fails to detect constitutes a special relationship between the initial condition and the parameters. As long as such relationship is established, there exists no admissible input, and the initial conditions need not to be at equilibrium. This case is demonstrated below.

Proposition 3. Given a generic initial condition $x_0 = [s_0, v_0]^T$ for the CTH-RV system (3.19), if $\tau = \frac{s_0}{v_0}$ and $k_2 = \frac{v_0}{s_0}$, no admissible input exists.

Proof. Please see the Appendix A.0.1 for details. □

Remark 4. Measuring the space gap alone $y(t) = s(t)$ leads to the same identifiability results as measuring both states $y(t) = [s(t), v(t)]^T$. One can arrive to this result by checking the rank of the observability-identifiability matrix $\mathcal{O}_I(\tilde{x}_0, u)$. This finding is in agreement with [169, 205] that using space gap instead of velocity or acceleration profile to calibrate a car-following model leads to sensitivity in the objective function.

Visualization As an illustration of the above analysis for CTH-RV, we visualize the output differences $e(\cdot, \cdot)$ corresponding the settings of Propositions 1-3. We first fix one parameter as θ_{true}, and compare the output error under another parameter θ. Let $e(\theta, \theta_{\text{true}})$ be defined as the *mean-squared-error* (MSE):

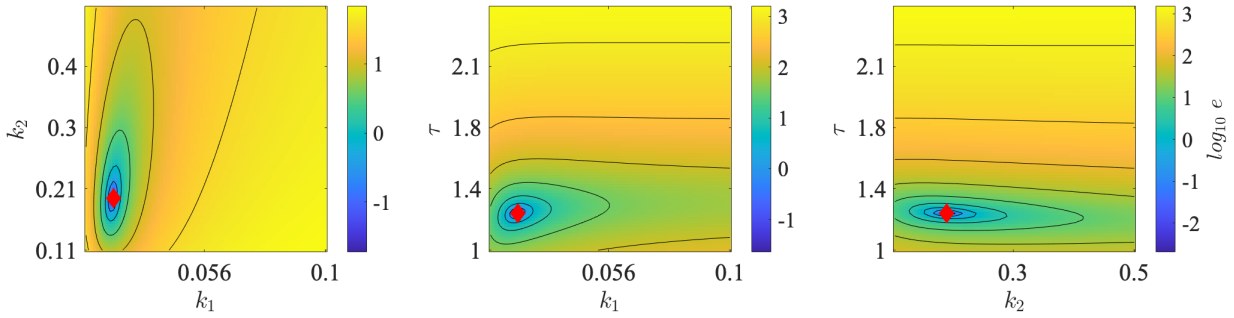
$$e(\theta, \theta_{\text{true}}) = \int_0^{\mathcal{T}} \|y(t, \theta_{\text{true}}) - y(t, \theta)\|_2^2 dt, \tag{3.24}$$

which describes the error between the true output simulated by θ_{true} and the output simulated by θ, under the same initial conditions and input. Let $\mathcal{T}$ denote the total time over which the ODE is solved.

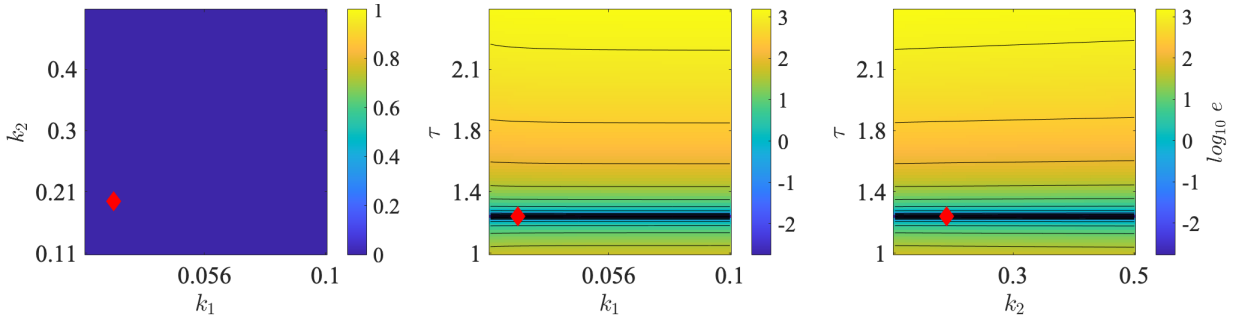
We visualize $e(\theta, \theta_{\text{true}})$ as one sweeps over the parameter space for CTH-RV (see Figure 3.2a-3.2c). Figure 3.2a shows that the error is zero only at $\theta = \theta_{\text{true}}$, indicating an identifiable system. Figure 3.2b and 3.2c, on the contrary, show that even for a structurally locally identifiable system, there exists an initial condition, parameters, and input combinations such that distinct parameters exist and produce the same output, as suggested by Proposition 2 and 3. Specifically, we see in Figure 3.2b that k_1 and k_2 are unidentifiable under an equilibrium initial condition and a constant input. This is consistent with the finding from [242]; in Figure 3.2c, we see that that k_1 is unidentifiable when the initial speed and space gap satisfies $s_0 = \tau v_0$, even though the initial condition is not at a equilibrium state (i.e., $v_0 \neq u_0$).

3.5.3. Structural local identifiability analysis for other models

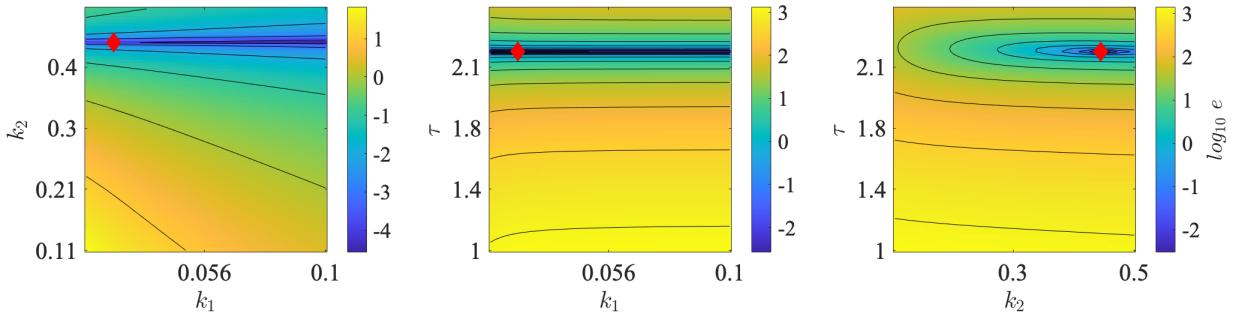
Following the same procedure as was used to analyze the CTH-RV, we check the observability-identifiability matrix $\mathcal{O}_I(\tilde{x}_0, u)$ for the other three models, i.e., the OV model (3.6), the FTL model (3.8) and the IDM (3.10). The structural identifiability results for each of the models under almost all initial conditions and the corresponding *a priori* set of admissible inputs are summarized



(a) An identifiable scenario: initial condition $x_0 = [60, 20]^T$, input $u(t) = 31$. $\theta_{\text{true}} = [k_1, k_2, \tau]^T = [0.0216, 0.1943, 1.2293]^T$.



(b) An unidentifiable scenario: equilibrium initial condition $x_0^* = [38.1, 31]^T$ and $u(t) = 31$. $\theta_{\text{true}} = [k_1, k_2, \tau]^T = [0.0216, 0.1943, 1.2293]^T$.



(c) An unidentifiable scenario: initial condition $x_0 = [72.7, 32.5]^T$ and $u(t)$ is shown in Figure 3.3. $\theta_{\text{true}} = [k_1, k_2, \tau]^T = [0.0216, 0.4472, 2.2361]^T$.

Figure 3.2: Visualization of $e(\theta, \theta_{\text{true}})$ for CTH-RV. Red diamond indicates θ_{true}. Even for structurally locally identifiable systems, there may exist an initial condition and input combination such that certain parameter become unidentifiable.

Table 3.1: Structural local identifiability summary.

Model f_{CF}	Parameters θ	Admissible polynomial input $u(t)$ degree n	
		Generic (almost all) x_0	Equilibrium x_0^*†
CTH-RV	k_1, k_2, τ	$n \geq 0$	$n \geq 1$
OV	α, a, b, h_m	$n \geq 0$	$n \geq 1$
FTL	C, γ	$n \geq 0$	N/A
IDM	a, b, s_j, v_f, T	$n \geq 0$	$n \geq 1$

† The equilibrium initial condition for each model is listed in Section 3.3.

in Table 3.1. For specific initial conditions (e.g., an equilibrium initial condition), the system may require a higher-order input to enable structural identifiability (also see Table 3.1).

Table 3.1 suggests that all the models are structurally locally identifiable, i.e., parameters for each model can be uniquely identified under any admissible input given almost all initial conditions, whether it is known or not, even if only space-gap is being measured. Moreover, a constant input degree $n = 0$ is an admissible input to enable structural identifiability of all the models. Additionally, all inputs $u(t)$ with degree $n \geq 0$ are also admissible to enable structural identifiability.

However, the analysis does not preclude the existence of an initial condition (e.g., an equilibrium initial condition) for which the models become unidentifiable for a constant input. For some models, higher-order inputs are required to enable structural identifiability. For CTH-RV, OV and IDM, any input with degree $n \geq 1$ is an admissible input to enable identifiablity for the equilibrium initial condition. On the other hand, the FTL completely loses identifiability given an equilibrium initial condition $v_0 = u_0$, since no admissible inputs can be found to enable identifiability.

Overall, the main outcome of the structural local identifiability analysis is that we find no fundamental intrinsic structural problems in any of the analyzed models. This means it is possible to design an experiment (i.e., chose an initial condition and an input) in order to uniquely infer the unknown model parameters.

The structural local identifiability analysis provides theoretical results that are generally valid for almost all numerical values of the initial conditions and the parameters, but the test could fail to detect unidentifiable scenarios (even for a simple CTH-RV model, it fails to detect a special initial condition (Proposition 2) or parameter dependency (Proposition 3) that causes non-identifiability). In practice, identifiability would be impossible when close to these special conditions (because of the rank deficiency of the observability-identifiability matrix). The probability of being in a small open set around these measure-zero sets, although unlikely, is not zero. Therefore, it is important to ask a different question in practice: *given the experimental design (i.e., given a specific initial condition and input trajectory that are known), whether there exists multiple distinct parameters that would produce the same output.* If multiple distinct parameters are indistinguishable in the output space, then the corresponding model(s) are unidentifiable. We proceed to numerical methods in the next section to address this question.

Table 3.2: Parameter bounds.

Models	Parameter θ	Lower bound $\theta_{\mathbf{min}}$	Upper bound $\theta_{\mathbf{max}}$
CTHRV	$[k_1, k_2, \tau_h]$	[0.001, 0.01, 0.1]	[1, 1, 3]
OV	$[\alpha, a, h_m, b]$	[0.5, 10, 2, 18]	[3.3, 32, 30, 45]
FTL	$[C, \gamma]$	[100, 1]	[600, 3]
IDM	$[s_j, v_f, T, a, b]$	[3, 21, 0.1, 0.1, 0.5]	[25, 41, 3, 3, 5]

3.6. **Practical identifiability analysis**

Although the initial condition and parameter subspace that loses structural identifiability for CTH-RV can be analytically solved for, it is much challenging to exhaustively find the special cases for other models composed of irrational functions. To this end, we use a numerical method to analyze a more specific notion of identifiability: *given a specific initial and input condition that are known and non-trivial, i.e., non-equilibrium initial condition and time-varying input, are there distinct parameters that produce the same or similar output?* If such distinct parameter sets are found, then the corresponding model is unidentifiable to this specific experimental setup, irrespective of the cause (initial condition or parameters fall into a measure zero set or the input is not informative enough). In this section, we present identifiability analysis using *direct test* for models (3.3), (3.6), (3.8) and (3.10). First, we provide the details of the numerical experiment setup, including a specific optimization formulation. Next, we solve (3.18) with 1) the tightest constraint $\epsilon = 10^{-6}$ and 2) the relaxed constraint $\epsilon > 10^{-6}$. Because the choice of the cut-off threshold ϵ is arbitrary, we incrementally increase ϵ to observe the sensitivity of model calibration to the measurement error.

For δ-identifiability at both $\epsilon = 10^{-6}$ and $\epsilon > 10^{-6}$, we first provide the detailed analysis on the CTH-RV model (3.3), and summarize the results for the others.

3.6.1. **Implementation details**

Consider the following optimization problem formulation for (3.18):

$$\begin{aligned} \underset{\theta_{1,i},\theta_{2,i}\in[\theta_{\min,i},\theta_{\max,i}],\forall i}{\text{maximize}} \quad & d(\theta_1,\theta_2) = \frac{1}{\sqrt{n_\theta}}\sqrt{\sum_{i=1}^{n_\theta}\left(\frac{\theta_{1,i}-\theta_{2,i}}{\theta_{\max,i}-\theta_{\min,i}}\right)^2} \\ \text{subject to} \quad & e(\theta_1,\theta_2) = \frac{1}{K}\sum_{k=0}^{k=K}\|y_k(\theta_1)-y_k(\theta_2)\|_2^2 \le \epsilon. \end{aligned} \tag{3.25}$$

The subscript i on θ denotes the i^{th} element in the parameter vector. The resulting distance (objective function) is a normalized Euclidean distance scalar between 0 and 1 bounded by $\theta_{\min}$ and $\theta_{\max}$ (Table 3.2), i.e., $d(\theta_{\min}, \theta_{\max}) = 1$. The trajectory difference $e(\cdot,\cdot)$ in the constraint is a mean squared error to measure the difference between $y_k(\theta_1)$ and $y_k(\theta_2)$, i.e., the discrete-time space-gap simulated with an Euler method using θ_1, θ_2, respectively. The number of decision variables is $2\times n_\theta$ where n_θ is the dimension of the parameters for each model. Recall that we denote the solution to (3.18) as δ^* and the corresponding parameter pair as (θ_1^*, θ_2^*).

The optimization solver used to solve (3.25) is `patternsearch` in the MATLAB global optimization

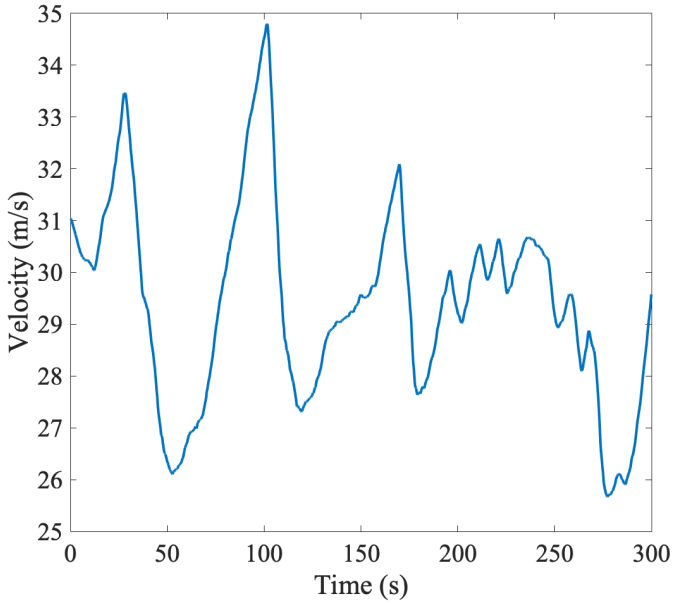


Figure 3.3: A time-varying lead vehicle velocity profile.

Table 3.3: Direct test finds two indistinguishable parameter sets for CTH-RV with ϵ =1e-6

Model	Parameter	θ_1^*	θ_2^*	δ^*
CTHRV	k_1	1	0.001	0.5774
	k_2	0.4472	0.4472	
	τ	2.236	2.236	

toolbox [139]. The `patternsearch` is a gradient-free optimization solver that works with potentially non-smooth objective functions. More details regarding the solver can be found in [9].

3.6.2. **Practical identifiability with error-free measurement**

In the numerical setting, we specify the tightest constraint at approximately zero (specifically, $\epsilon = 10^{-6}$) and solve for the optimization problem (3.25) with a time-varying input profile (Figure 3.3) and a fixed, non-equilibrium initial condition $x_0 = [72.7, 32.5]^T$. We use this initial condition and input for all of the models in the analysis for the remaining of this chapter. Given this specific experiment setup, direct test aims to find indistinguishable (thus unidentifiable) parameters that separate farthest in the parameter space. This specific initial condition is chosen to illustrate the utility of the direct test, i.e. given an arbitrary but non-equilibrium initial condition, can we find two distinct parameters that generate the same output.

CTH-RV The direct test result for CTH-RV by solving the optimization problem with $\epsilon = 10^{-6}$ is listed in Table 3.3. Table 3.3 suggests that given a non-equilibrium lead velocity profile (Figure 3.3) and an arbitrary, non-equilibrium initial condition that are known, the CTH-RV is unidentifiable. Solving problem (3.25) results in $\delta^* = 0.577$, meaning there exists two distinct parameters that

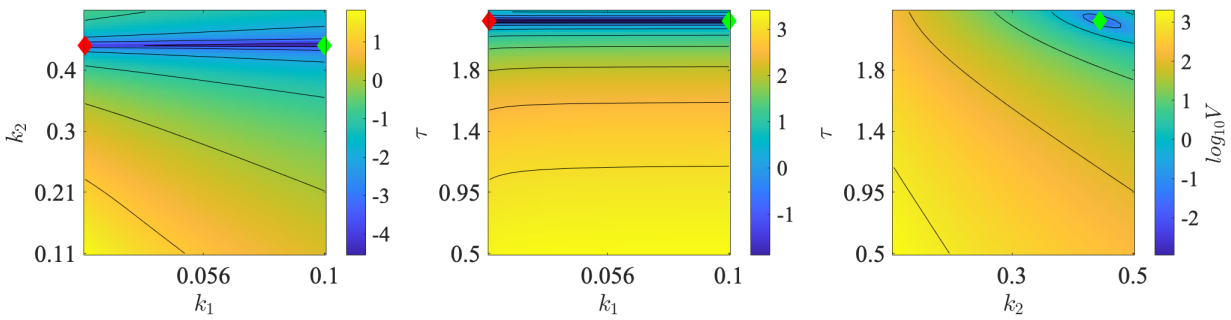


Figure 3.4: Visualize indistinguishable parameter sets θ_1^* (red diamond) and θ_2^* (green diamond) from Table 3.3.

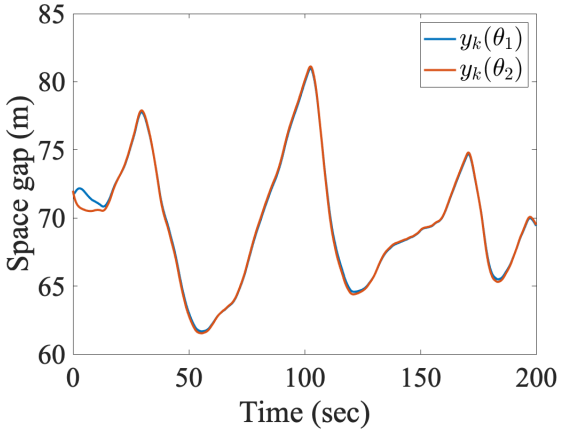


Figure 3.5: Visualize practically indistinguishable parameter sets for CTHRV.

generate the same output. Figure 3.4 visualizes this result: θ_1^* and θ_2^* are far apart but produce the same output. The value of $e(\theta, \theta_1^*)$ remains constant along the unidentifiable parameter. Furthermore, the unidentifiable parameters fall into a set such that $\tau = s_0/v_0$ and $k_2 = v_0/s_0$. This corresponds to Proposition 3 where k_1 cannot be uniquely identified.

Additionally, we discovered that when the initial conditions are in a neighborhood of the measure-zero set identified in Proposition 3, CTH-RV can still be practically unidentifiable, as shown in Figure 3.5. In this example, the initial condition is $s_0 = 72$, $v_0 = 32$, and the parameters are $\theta_1 = [1, 0.394, 2.35]^T$ and $\theta_2 = [0.1, 0.394, 2.35]^T$. A small output difference is obtained ($\epsilon = 0.0745$) with distinct parameters ($\delta^* = 0.5201$). Note that $k_2 \approx \dfrac{v_0}{s_0}$ and $\tau \approx \dfrac{s_0}{v_0}$, which mean that the initial conditions are not exactly in the measure-zero set as shown in Proposition 3, but in a neighborhood of a it.

Other models As for the other three models, the solution to the direct test (3.25) is summarized in Table 3.4. The results indicate that the FTL (3.8) is also unidentifiable under the specific experiment setup ($\delta^* = 0.677$). OV (3.6) and IDM (3.10) are practically identifiable ($\delta^* = 0.0117$ and 0.0476, respectively, both are small), which suggests that only parameters that are close to each other can produce the same output. The solutions found by numerical direct test are visualized in

Table 3.4: Direct test finds indistinguishable parameter sets for other models with ϵ =1e-6

Model	Parameter	θ_1^*	θ_2^*	δ^*
OV	α	3.0772	3.1427	0.0117
	a	19.7485	19.7451	
	h_m	22.2094	22.2122	
	b	23.2986	23.2915	
FTL	C	130.0285	599.9699	0.6766
	γ	1	1.3582	
IDM	s_0	10.5615	8.3913	0.0476
	v_0	35.788	35.771	
	T	2.787	2.903	
	a	2.559	2.494	
	b	3.395	3.395	

Figures 3.4-3.8 for the corresponding models.

3.6.3. **Practical identifiability when measurement noise is present**

The practical identifiability is nontrivial to quantify because 1) the threshold for δ^* to be identifiable/unidentifiable cannot be easily generalized across all models. Parameters for each model also have different physical meanings. 2) The trajectory error ϵ has multiple possible formulations (RMSE, ARE, MSE, etc.), making a universal cut-off threshold for practical identifiability impossible. Therefore, we provide the sensitivity analysis instead. The analysis is to provide a sense of robustness of parameter estimation with the presence of imperfect measurements (e.g., with measurement errors).

The value of ϵ used in the constraint of the direct test problem is user-defined, and must be specified before solving the numerical problem. Therefore, we vary ϵ values to find the maximum parameter distances for each ϵ (see, for example, $e(\cdot,\cdot) = 0, 0.1, 1$ and 5m^2 for IDM in Figure 3.9). This is to analyze the sensitivity of the maximum distance between two indistinguishable parameters with respect to the tolerance on the output differences. Previous studies suggest that calibrating on empirical data achieves the lowest MSE of 1-2m^2 in space gap [81, 242], our analysis on practical identifiability will focus on $e(\cdot,\cdot)$ to be less than 1m^2, or $\epsilon \in [0, 1]$.

In particular, we select ϵ equally-spaced in log space between 10^{-6} and 10^0, and solve for δ^* and the corresponding (θ_1^*, θ_2^*) in problem (3.25). The resulting curves shown in Figure 3.10 are the solutions δ^* as a function of ϵ in the constraint for each model. The curve is the maximum distance in parameter space such that the output error does not exceed ϵ. Clearly, as the output difference ϵ increases, so does the maximum parameter distance δ^*. Intuitively, given a specific initial condition and lead vehicle trajectory, as measurement error increases, the best-fit parameters are more likely to be non-unique, and the estimated parameters are less robust in the worst case.

Furthermore, Figure 3.10 suggests that the CTH-RV and the FTL are practically unidentifiable no matter what output error threshold to chose, under the given initial and input conditions. I.e.,

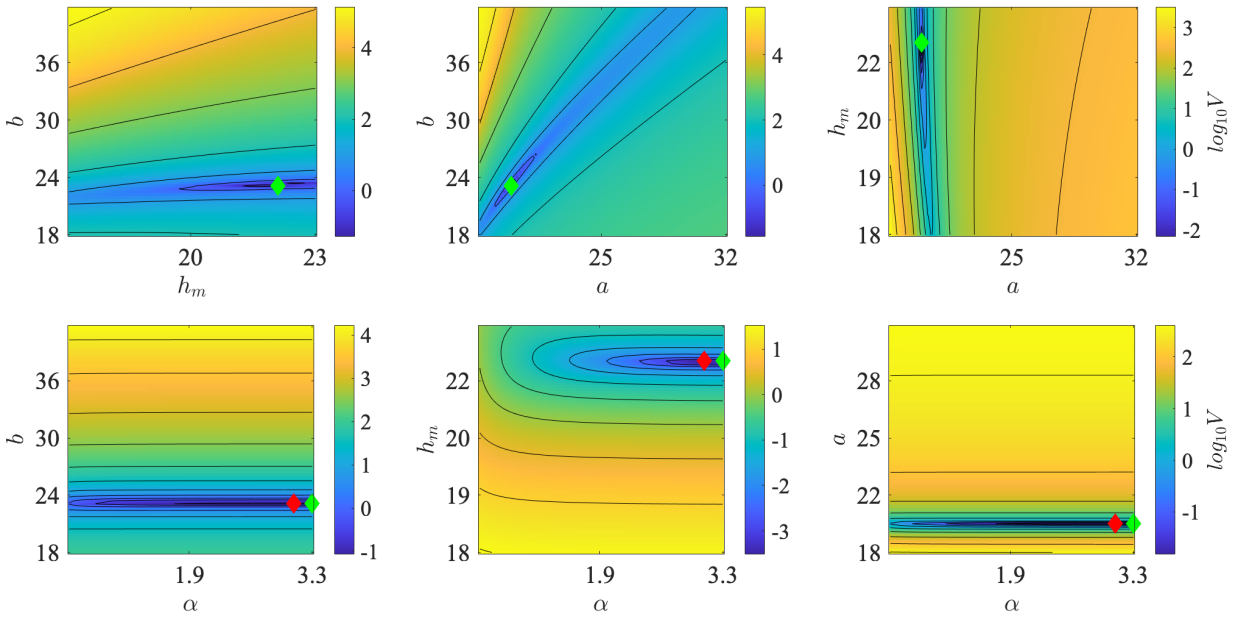


Figure 3.6: Visualize indistinguishable parameter sets θ_1^* (red diamond) and θ_2^* (green diamond) for OV from Table 3.4.

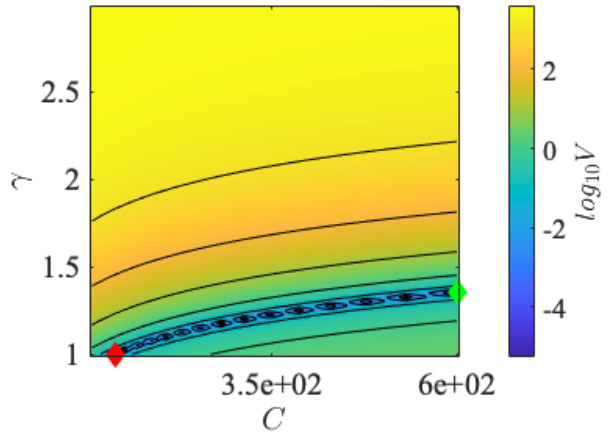


Figure 3.7: Visualize indistinguishable parameter sets θ_1^* (red diamond) and θ_2^* (green diamond) for FTL from Table 3.4.

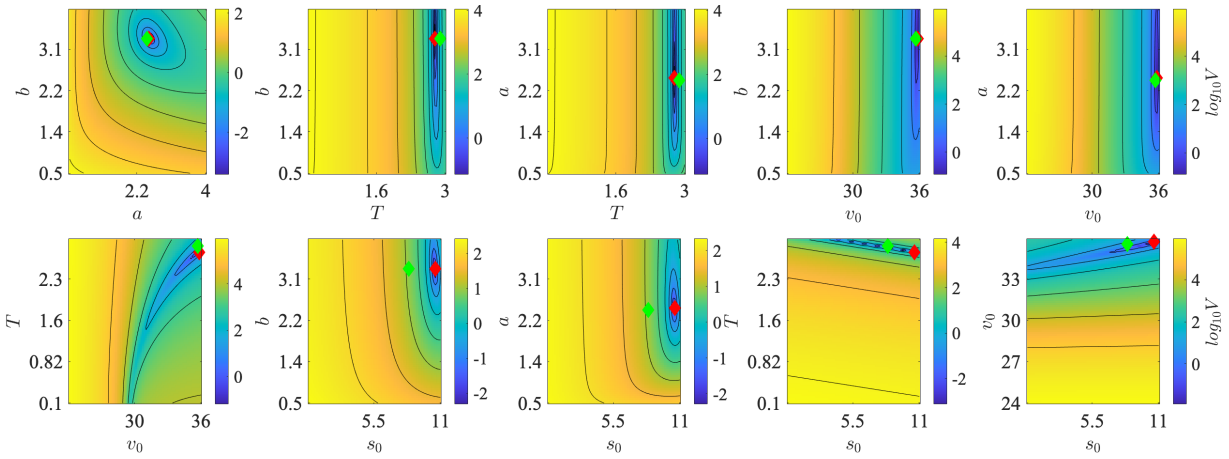
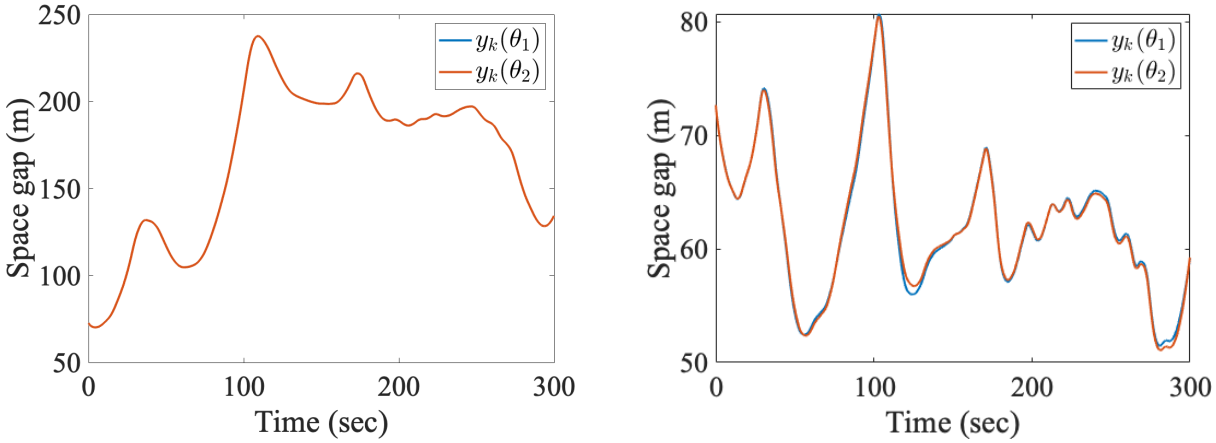


Figure 3.8: Visualize indistinguishable parameter sets θ_1^* (red diamond) and θ_2^* (green diamond) for IDM from Table 3.4.

there is no guarantee to find the unique parameters for the CTH-RV and the FTL even with perfect (error-free) data. On the other hand, the OV and the IDM can be practically identifiable when the threshold for output error is decreased. However, identifiability may not be achieved if the data is contaminated with noises (ϵ is large).

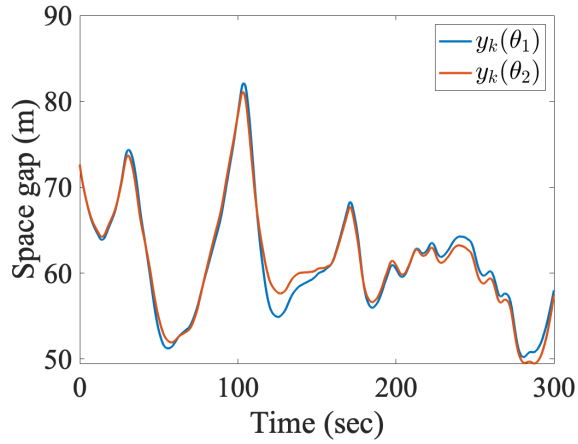
In summary, the *direct test* is a conceptually straightforward method to test identifiability given a particular experimental design (i.e., with a known initial condition and a known input). It is very helpful to know (for model that is structurally locally identifiable) if the experiment leads to practical unidentifiability. Our approach provides a tool that highlights where an experiment can fail in the sense that similar outputs are obtained by distinct parameters. This practical identifiability test is complementary to that of *structural identifiability* analysis, which provides theoretical identifiability results for almost all initial conditions and parameters, but does not preclude the existence of special initial conditions and parameters (those belong to a set of measure zero) that result in non-identifiability (shown in Proposition 2 and 3). Note also that the *direct test* does not aim to find *all* initial conditions and parameters that fall into a set of measure zero, but simply tests if indistinguishable parameters exist given a full experimental setup, which can be useful in practice. Having a user-defined output difference threshold ϵ gives the *direct test* more flexibility to understand the impacts of output measurement errors.

However, numerical test also has some disadvantages. For example, depending on the optimization solver and the set of starting points, the numerical approach may not find the true global maximum, and thus may misleadingly provide a lower bound on δ^*, rather than the global maximum. In addition, the numerical direct test is performed based on a specific initial condition and input trajectory that are known, which limits its use to inform experimental designs. This approach does not consider practical experimental design, including how to design one or several experiments to best estimate the parameters, which is an interesting direction for further investigation. Nevertheless, it is insightful as a verification tool to diagnose the potential unidentifiability under a specific setup.

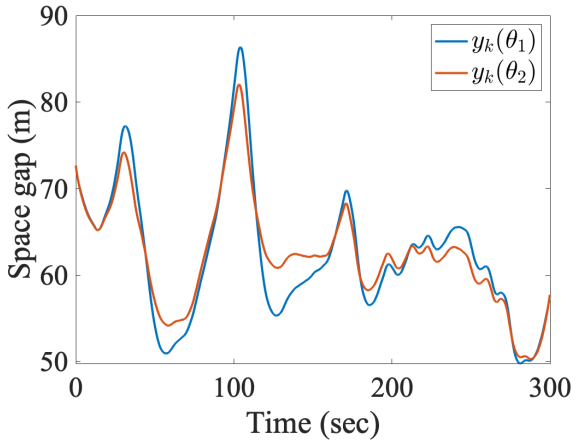


(a) $e(\theta_1, \theta_2) = 0$

(b) $e(\theta_1, \theta_2) = 0.1$



(c) $e(\theta_1, \theta_2) = 1$



(d) $e(\theta_1, \theta_2) = 5$

Figure 3.9: A demonstration of various trajectory differences $e(\theta_1, \theta_2)$ of IDM.

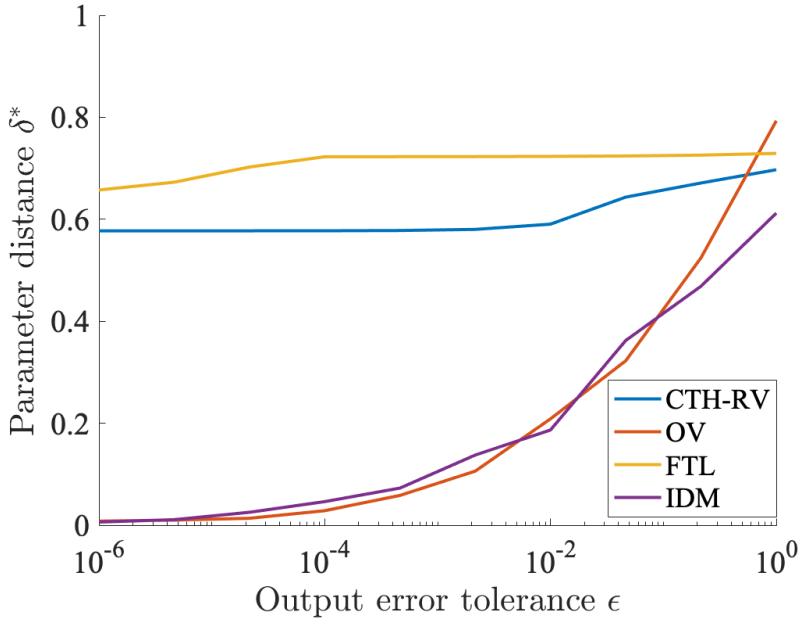


Figure 3.10: The sensitivity of practical identifiability with respect to the error threshold ϵ. The specific setup is $x_0 = [72.7, 32.5]^T$ and input $u(t)$ shown in Figure 3.3 for all the models.

3.7. **Conclusion**

In this work, we study the structural and practical identifiability of four car-following models.

The structural identifiability analysis is carried through using a *differential geometry* approach. It provides theoretical identifiability results under error-free assumptions for almost all initial conditions and parameters, and informs the admissible input condition that enables structural identifiability. However, it does not preclude the existence of initial conditions and parameters that lose identifiability. The results from the *differential geometry* analysis show that all models are structurally locally identifiable, i.e., all models in theory have unique parameterizations when fitting with space gap data given almost all initial conditions and any admissible input. For some initial conditions such as an equilibrium initial condition, higher-order input is required for CTH-RV, OV and IDM to be identifiable. No admissible input exists for FTL when the initial condition is at equilibrium to enable identifiability.

As a complementary analysis, we use a numerical *direct test* to study practical identifiability given a specific experimental setup (a given known initial condition and input). It also provides insights on the sensitivity of parameter identifiability due to measurement errors on the output. The *direct test* finds that CTH-RV and FTL are practically unidentifiable, meaning that there exists multiple distinct parameters that produce the same output, given a specific experimental setup. It also suggests that OV and IDM are practically identifiable when the measurement error is small. Although only four models are investigated, the provided methods can be applied to other car following models.

The findings also open up questions for future research. For example, we are interested to design an experiment or sets of experiments that avoid problematic initial conditions for identifiability. Design

of reduced order models that allow robust parameter identification while reflecting the dynamics of car following behavior are also of interest.

CHAPTER 4

Applications on adaptive cruise control system identification

4.1. Introduction

As traffic composition is becoming increasingly diverse due to the rising penetration of vehicle automation, identification of individual driving behavior becomes more important to understand, for example, driving comfort, safety, and its broader implication on the traffic at larger-scale. We are facing many new challenges: growing market demands for a wide range of real-time application requires a prompt and high-fidelity estimation of individual driver-vehicle dynamics. We are also provided with new opportunities. For example, on-board acquisition equipment provides continuous and high-frequency signals that contain rich contextual information; the prevalence for cloud and edge computing offers seamless connection with the physical systems. Because of these reasons, efficient algorithms for identification of the individual driving behavior is desired.

The impact of individual driver-vehicle unit on the broad traffic patterns can be studied via the so called "string stability" property of the car-following system. The interpretation of a string stable vehicle is that a homogeneous platoon consisting of these vehicles will dissipate disturbances rather than amplify them as the perturbation propagates in through platoon. Although string stability manifests along a platoon of vehicles, it is a property of the individual vehicle car following behavior [253, 153]. Thus identifying the car-following model parameters of the follower vehicle is sufficient to analytically prove the string stability of the follower vehicle. One practical use case of studying parameter identification and string stability is the design of string stable ACC systems, which has been an important topic in the vehicle control community for decades [174, 96, 23, 127, 200]. The ACC system enables the vehicle (instead of the human driver) to adjust velocity in response to the vehicle ahead. Recently, as the vehicle systems have transitioned from research to practical deployments on commercial vehicles, the traffic modeling community is now in need of more efficient system identification routines to understand how these vehicles behave in practice. Surprisingly, all commercial systems that have been tested [137, 136, 194, 79, 110, 146] are shown to be string unstable and with varying performance characteristics. Being able to characterize the behavior of the ACC system in real time has implications for traffic management, where an emerging area of research [79, 256, 269, 235, 49] aims to dampen phantom traffic jams caused by string unstable driving behavior [196].

This chapter considers the problem of estimating the parameters of adaptive cruise controlled vehicles using online algorithms that can sequentially estimate the parameters when new measurements become available. Two online methods are used based on *recursive least squares* (RLS) and *particle filtering* (PF), and both are shown to provide accurate estimates. As a proof of demonstration, we also implement the methods on data collected from a 2019 stock SUV with ACC, using only data from the vehicle's existing on-board sensors.

The new contributions to this chapter are:

- **The development of online parameter estimation algorithms** to solve the problem of recovering adaptive cruise control parameters. We use two online estimation algorithms, RLS and PF, that are fast and scalable for real time system identification of ACC dynamics.
- **Provide a parameter identifiability analysis of the methods**. We provide an analysis

of the parameter identifiability of both methods to understand if and when the estimates can be theoretically recovered. This analysis is important but missing for the batch methods previously applied to estimate parameters of ACC systems.

- **Experiment demonstration** Finally, we provide numerical and real world examples illustrating the performance of the methods in controlled numerical simulations, as well as on a modern ACC vehicle. The real vehicle experiment uses only the onboard CAN bus data, which provides a novel experimental approach to understand the behavior of ACC vehicles.

The remainder of the chapter is organized as follows. Section 4.2 reviews the ACC model and outlines the batch optimization method as a benchmark for parameter estimation. Section 4.3.1 introduces the online RLS and PF based estimators, and provides an analysis of the estimators at equilibrium driving conditions. Section 4.4 demonstrates the performance of the estimation routines on synthetically generated (simulated) data, in order to assess the performance of the methods under controlled settings. Section 4.5 addresses the practical performance of the method using data from a real vehicle platform. It presents both the experimental protocol for collecting ACC radar and speed measurements directly from a 2019 vehicle's CAN bus, and the results of parameter estimation on that data. Finally, Section 4.6 explores future research directions.

4.2. Preliminaries

In this section, we briefly review a common model assumed for ACC vehicle dynamics, and then review a standard simulation-based optimization method to estimate the model parameters used in this work as a benchmark.

4.2.1. Model description and string stability

With the increasing interest in how vehicles with automated driving systems [96, 23] will affect traffic flow patterns, several works have looked at modeling ACC vehicles using car-following models [146, 80, 79]. A common variation of these models is the *constant time headway relative velocity* (CTH-RV) model (3.3). Here we rewrite it as:

$$\dot{v}(t) = f(\boldsymbol{\theta}, s(t), v(t), \Delta v(t)) = \alpha(s(t) - \tau v(t)) + \beta(\Delta v(t)), \tag{4.1}$$

where s, v, and Δv are the space gap, velocity, and velocity difference between the ACC vehicle and a leading vehicle. The vector of model parameters $\boldsymbol{\theta} = [\alpha, \beta, \tau]^T$ control the gain on the constant time headway term and the relative velocity term respectively, while the parameter τ is the time gap at equilibrium.

We note that models considering constant time headway and relative velocity terms are regularly used both to design string stable adaptive cruise control systems, as well as to model the behavior of vehicles under ACC control [147, 146, 127, 259, 260, 15]. Compared to other modeling choices, it is observed that CTH-RV model performs about as well in terms of data fitting real ACC systems compared to more complex nonlinear models [81]. However, the model is a simplification of the proprietary control logic and complex vehicle dynamics of real ACC vehicles, and the quality of fit can drop for some specialized vehicles (e.g., hybrid vehicles) [79]. To avoid the need to know the proprietary control logic, we adopt a similar strategy to what is done for human drivers, namely model the full system as an ordinary differential equation. For the remainder of this work, we adopt the CTH-RV as the assumed model of the ACC equipped vehicle.

Given (4.1), it is easy to check the string stability [253] of the ego (i.e., follower) vehicle by evaluating partial derivatives of the model with respect to s, v, and Δv. Following the analysis of [153], if

$$\alpha^2\tau^2 + 2\alpha\beta\tau - 2\alpha \geq 0, \tag{4.2}$$

then the model is said to be $\mathcal{L}_2$ *strict string stable*, which is consistent with the string stability condition provided in [253]. Moreover, if

$$(\alpha\tau + \beta)^2 - 4\alpha \geq 0, \tag{4.3}$$

then the model is said to be $\mathcal{L}_\infty$ *strict string stable* [153]. All studies that have collected empirical data on commercial ACC systems have found them to be string unstable [80, 79, 146, 110] (in the $\mathcal{L}_2$ sense). In Section 4.5, we illustrate that the new online methods introduced in this work find that a stock 2019 SUV is neither $\mathcal{L}_2$ nor $\mathcal{L}_\infty$ strict string stable.

Note that although string stability manifests along a platoon of vehicles, it is a property of the individual vehicle car following behavior [253, 153]. Thus identifying the car-following model parameters of the follower vehicle is sufficient to analytically prove the string stability of the follower vehicle. The interpretation of a string stable vehicle is that a homogeneous platoon consisting of these vehicles will dissipate disturbances rather than amplify them as the perturbation propagates in through platoon.Since no priori knowledge of the ACC system string stability is assumed in the experiments, we do not use string stability as constraints during parameter estimation.

4.2.2. **Offline batch optimization**

Here a well known batch technique for car-following parameter estimation [169, 80] is reviewed to estimate the parameters of the *ordinary differential equation* (ODE) model (4.1). The parameter estimation problem is posed as an optimization problem in which the ACC model appears as a constraint. It can be directly solved as a simulation-based optimization problem using standard descent-based optimization routines.

The parameter values are optimized to minimize the *root mean squared error* (RMSE) between simulated space gap data and recorded space gap data. The RMSE space gap error is used here because it was found to perform well in previous works [169, 105]. The general form of optimization problem is written as:

$$\begin{aligned} \text{minimize}: \quad & \sqrt{\tfrac{1}{T}\int_0^T (s^{\mathrm{m}}(t) - s(t))^2 dt} \\ \text{subject to:} \quad & \dot{s}(t) = u(t) - v(t) = \Delta v \\ & \dot{v}(t) = f(\boldsymbol{\theta}, s, v, \Delta v), \end{aligned} \tag{4.4}$$

with possible additional constraints on the initial conditions, and bounds on the parameters. In (4.4), $f(\boldsymbol{\theta}, s, v, \Delta v)$ corresponds to the car-following model in (4.1). The term $u(t)$ is the lead vehicle velocity as a function of time and is assumed to be available from measured data. Similarly, $s^{\mathrm{m}}(t)$ denotes the measured space gap, which is compared to the space gap predicted by the model in the objective function. The total time of the dataset and simulation is T.

It is important to note that the problem is nonlinear in the decision variables (the state and model parameters), and depending on the form of the car-following model, it may also be non-convex. To

combat this potential problem the optimization problem can be solved many times, with each run starting from randomly selected different initial candidate parameter values, as in [169].

4.3. **Online parameter estimation techniques**

We next introduce two online methods to estimate the adaptive cruise control model parameters using velocity, space gap, and relative velocity data.

4.3.1. **Recursive least-squares formulation**

First we derive a RLS estimator. Unlike (4.4), the least-squares method proposed here does not require multiple starting points or repeatedly solving an ODE within each optimization run, substantially reducing the runtime. We briefly derive the least-squares formulation for the ACC car-following model (4.1).

First we rewrite the continuous time ODE (4.1) in discrete-time using a forward Euler step scheme:

$$v_{k+1} = v_k + \alpha(s_k - \tau v_k)\Delta T + \beta(u_k - v_k)\Delta T, \tag{4.5}$$

where v_k, s_k and u_k denote the velocity of the follower vehicle, the space gap, and the velocity of the leading vehicle at timestep k, respectively. The term ΔT is the timestep size, which is selected to correspond to the frequency at which the velocity, space gap, and relative velocity data is measured (e.g., on the order of $1/10$ of a second for some sensor platforms including the experiments presented later in this work). The dynamics can be rewritten as:

$$v_{k+1} = \gamma_1 v_k + \gamma_2 s_k + \gamma_3 u_k, \tag{4.6}$$

with $\gamma_1 := (1 - (\alpha\tau + \beta)\Delta T)$, $\gamma_2 := (\alpha\Delta T)$ and $\gamma_3 := (\beta\Delta T)$. Note that instead of directly estimating the parameters α, β, τ, we can instead estimate $\gamma_1, \gamma_2, \gamma_3$. Except in the degenerate case when $\gamma_2 = 0$, we can always uniquely determine the values of α, β, τ given a set of values for $\gamma_1, \gamma_2, \gamma_3$.

We now demonstrate that one can recover $\boldsymbol{\gamma} := [\gamma_1, \gamma_2, \gamma_3]^T$ from an experimental dataset containing (v_k, s_k, u_k) for all $k \in \{1, ..., K\}$, via least-squares. We expand (4.6) in time by stacking the uniformly sampled measurements to obtain:

$$\begin{bmatrix} v_2 \\ v_3 \\ \vdots \\ v_K \end{bmatrix} = \begin{bmatrix} v_1 & s_1 & u_1 \\ v_2 & s_2 & u_2 \\ \vdots & \vdots & \vdots \\ v_{K-1} & s_{K-1} & u_{K-1} \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \gamma_3 \end{bmatrix}, \tag{4.7}$$

or

$$Y = X\boldsymbol{\gamma}. \tag{4.8}$$

The term Y contains the values of v_k from timestep 2 to K. The term X contains measurements of v_k, s_k, and u_k from timestep 1 to $K - 1$ in column-wise order.

Given the data matrices Y and X, $\boldsymbol{\gamma}$ has a unique solution if and only if $\text{rank}(X) = \text{rank}([X \quad Y]) = 3$. Note that this condition is not satisfied at equilibrium, where $v_i = v_j = u_k$ for all timesteps i, j and k. In other words, using only data from equilibrium driving, it is not possible to recover

the model parameters. In non-equilibrium driving and when sensor noise is present, it is easy to generate an over determined system of equations, motivating the search for least squares solution to (4.7).

To convert the least squares problem into an online method, a recursive implementation is desired. The least squares solution to (4.7) has an exact recursive implementation by considering the k^{th} row of measurements $\{Y_k, X_k\}$ one row at a time. The least squares estimate of the parameter vector at time k using all data collected from timestep 1 through k, denoted $\hat{\boldsymbol{\gamma}}_k$, can be sequentially updated by:

$$\hat{\boldsymbol{\gamma}}_k = \hat{\boldsymbol{\gamma}}_{k-1} + P_k X_k (Y_k - \hat{\boldsymbol{\gamma}}_{k-1} X_k), \tag{4.9}$$

where $P_k^{-1} = \sum_{i=1}^{k} X_i X_i^T$ is the cumulative outer product of X_k. Solving (4.9) requires an initial estimate of the parameters $\boldsymbol{\gamma}_0 \sim \mathcal{N}(\hat{\boldsymbol{\gamma}}_0, P_0)$, which are specified in the numerical experiment in Section 4.4.

4.3.2. Online joint state and parameter estimation formulation

The parameter estimation problem can also be framed as an online joint parameter and state estimation problem, in which model and measurement noises are explicitly considered. Such methods, if fast enough, may also be used for real-time processing of data in order to estimate the model parameters during data collection.

Problem formulation

To jointly estimate the state and model parameters, we consider an augmented state formulation in which the model parameters are added to the state vector. The model of the evolution of the augmented state is completed by assuming the parameters are constant in time.

We proceed as follows. First, $\boldsymbol{\theta} = [\alpha, \beta, \tau]^T$, is concatenated to the physical system state $\boldsymbol{x}_k \in \mathbb{R}^2 = [s_k, v_k]^T$ to form an augmented state $\boldsymbol{x}_k^a \in \mathbb{R}^5$. This is written as:

$$\boldsymbol{x}_k^a = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{\theta} \end{bmatrix}_k = \begin{bmatrix} s & v & \alpha & \beta & \tau \end{bmatrix}_k^T. \tag{4.10}$$

The discrete time dynamics of the augmented system using the same discretization approach as (4.5) can be written as:

$$\boldsymbol{x}_k^a = \mathcal{F}_d(\boldsymbol{x}_{k-1}^a, u_{k-1}), \tag{4.11}$$

where $\mathcal{F}_d$ refers the system dynamics in the augmented state. The nonlinearities in the dynamics appear due to the product of augmented state variables representing the ACC model parameters and the physical states.

The augmented state dynamics (4.11) are written as:

$$\mathcal{F}_d(\boldsymbol{x}^a_{k-1}, u_{k-1}) = \begin{bmatrix} s_{k-1} + \Delta T(u_{k-1} - v_{k-1}) \\ v_{k-1} + \Delta T\left[\alpha_{k-1}(s_{k-1} - \tau_{k-1}v_{k-1}) + \beta_{k-1}(u_{k-1} - v_{k-1})\right] \\ \alpha_{k-1} \\ \beta_{k-1} \\ \tau_{k-1} \end{bmatrix} \tag{4.12}$$

Additionally, a measurement equation is written to reflect the condition that the physical states (s_k, v_k) may be directly measured, but we do not measure the parameters $(\alpha_k, \beta_k, \tau_k)$:

$$\boldsymbol{y}_k = C\boldsymbol{x}^a_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{x}^a_k. \tag{4.13}$$

In (4.13), $y_k \in \mathbb{R}^2$ are the measurements, and C is the measurement matrix.

Particle filter

Because the augmented state (of the nonlinear augmented system) is to be estimated, a nonlinear estimator must be considered. Here, we outline an approach to estimate the augmented state using a PF.

The filter takes in the discrete-time system that considers model and measurement noises. The state-space form is written as:

$$\begin{aligned} \boldsymbol{x}^a_k &= \mathcal{F}_d(\boldsymbol{x}^a_{k-1}, u_{k-1}) + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= C\boldsymbol{x}^a_k + \boldsymbol{\nu}_k, \end{aligned} \tag{4.14}$$

where $\boldsymbol{w}_k \sim (0, Q) \in \mathbb{R}^5$ and $\boldsymbol{\nu}_k \sim (0, R) \in \mathbb{R}^2$ are independent white noise processes for the model and the measurement equations, respectively, at time k. $Q \in \mathbb{R}^{5\times5}$ and $R \in \mathbb{R}^{2\times2}$ are the known process and measurement error covariance matrices.

Recall that the Bayesian state estimation method sequentially approximates the posterior *probability density function* (PDF) of the augmented state at step k given past observations, i.e., $p(\boldsymbol{x}^a|\boldsymbol{y}_{1:k})$. The Bayesian state estimation method can be summarized into two parts:

1. **State propagation**: obtain the prior distribution at k:

$$p(\boldsymbol{x}^a_k|\boldsymbol{y}_{1:k-1}) = \int p(\boldsymbol{x}^a_k|\boldsymbol{x}^a_{1:k-1})p(\boldsymbol{x}^a_{k-1}|\boldsymbol{y}_{1:k-1})d\boldsymbol{x}^a_{k-1} \tag{4.15}$$

2. **State update**: obtain the posterior distribution at k:

$$p(\boldsymbol{x}^a_k|\boldsymbol{y}_{1:k}) = \frac{p(\boldsymbol{y}_k|\boldsymbol{x}^a_k)p(\boldsymbol{x}^a_k|\boldsymbol{y}_{1:k-1})}{p(\boldsymbol{y}_k|\boldsymbol{y}_{1:k-1})}. \tag{4.16}$$

The particle filter [51, 38], among other filtering techniques, is deployed to approximate the prior and the posterior distributions from equations (4.15) and (4.16) because of its flexibility in noise distribution and its relaxed assumption about the linearity of the (augmented) dynamics of the

Algorithm 1 Particle filter

Initialize $(k = 0)$

Draw i particles $\{\boldsymbol{x}_0^{a,(i)}\}_{i=1:N_p}$ from an initial distribution $p(\boldsymbol{x}_0^a)$. Assign equal weights $\omega_0^{(i)} = 1/N_p$, where $i = 1, \ldots, N_p$, and N_p is the number of particles.

for $k = 1 \ldots T$ **do**

State propagation:

$\boldsymbol{x}_k^{a,(i)} = \mathcal{F}_d(\boldsymbol{x}_{k-1}^{a,(i)}, u_{k-1}) + \boldsymbol{w}_k^{(i)}$ for all i.

State update:

Assign weight: $\omega_k^{(i)} \coloneqq \omega_{k-1}^{(i)} p(\boldsymbol{y}_k | (\boldsymbol{x}_k^{a,(i)})$ for all i.

Normalize weight: $\omega_k^{(i)} \coloneqq \omega_k^{(i)} / \sum_{i=1}^{N_p} \omega_k^{(i)}$ for all i.

Resample:

Draw $\boldsymbol{x}_k^{a,(i)}$ with probability $\omega_k^{(i)}$ for all i.

end

system. PF uses weighted particles (samples) to approximate the conditional state distribution given all measurements up to the current timestep using a sequential estimation approach. Therefore, the output is a probability distribution for each parameter at each time step.

A summary of the PF is written in Algorithm 1. During implementation, it is important to monitor the effective particle size [199] to ensure a valid estimation result. For more details on the PF implementation, readers are referred to standard references such as [188].

4.3.3. **Observability analysis**

In this section we provide insights on the ability to estimate the ACC model parameters via an observability analysis. An observable system indicates theoretically that its initial state can be inferred from observing the outputs. For parameter estimation in the joint state-parameter form, recovering the initial state indicates identifying the non-changing parameters. In this work we consider a special case of parameter observability if the parameters are considered as constant state variables. This assumption allows us to equate the notion of observable augmented state to uniquely identifiable parameters given measurements. Given that the augmented state dynamics (4.14) are nonlinear, observability must be assessed on a linearized version of the model. This can only be done at fixed values of the augmented state, and is explained as below.

First we write the linearized state-space model of the nonlinear discrete-time system (4.11):

$$\begin{aligned} \boldsymbol{x}_k^a &= A_{k-1} \boldsymbol{x}_{k-1}^a + B_{k-1} u_{k-1} \\ \boldsymbol{y}_k &= C \boldsymbol{x}_k^a, \end{aligned} \tag{4.17}$$

where A_k is the Jacobian of $\mathcal{F}_d$ defined in (4.11) with respect to the augmented state variables at time k, B_k is the Jacobian with respect to the control inputs at time k, and C is the time-invariant

measurement matrix as defined above in (4.13). Further, A_k can be written as

$$A_k = \frac{\partial \mathcal{F}_d}{\partial \boldsymbol{x}^a}\Big|_{\boldsymbol{x}_k^{a*},u_k^*} \begin{bmatrix} 1 & -\Delta T & 0 & 0 & 0 \\ \alpha\Delta T & 1-\alpha\tau\Delta T-\beta\Delta T & (s-\tau v)\Delta T & (u-v)\Delta T & -\alpha v\Delta T \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{\boldsymbol{x}_k^{a*},u_k^*}, \quad (4.18)$$

where $\boldsymbol{x}_k^{a*}, u_k^*$ are the state and input points about which the system is linearized and B_k is defined similarly.

We choose to analyze the system observability by computing the above partial derivatives evaluated at an equilibrium point. The condition for equilibrium reduces to zero acceleration and space gap change, i.e.:

$$\begin{aligned} u_k - v_k &= 0 \\ s_k - \tau_k v_k &= 0\ . \end{aligned} \quad (4.19)$$

In addition, the system (4.11) reduces to a linear time invariant system, and A_k from (4.18) at equilibrium simplifies to:

$$A = \begin{bmatrix} 1 & -\Delta T & 0 & 0 & 0 \\ \alpha\Delta T & 1-\alpha\tau\Delta T-\beta\Delta T & 0 & 0 & -\alpha v\Delta T \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

From here the observability matrix can be calculated using A and C as follows:

$$\mathcal{O} = \begin{bmatrix} C,\ CA,\ CA^2,\ CA^3,\ CA^4 \end{bmatrix}^T,$$

where $\mathcal{O}$ is the observability matrix, the rank of which is used to assess observability.

When analyzed at any equilibrium point, the resulting observability matrix has $\text{rank}(\mathcal{O}) = 3 \neq 5$, corresponding to a non-observable system. The corresponding null space of the observability matrix is:

$$\text{null}(\mathcal{O}) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T, \quad (4.20)$$

indicating two unidentifiable parameters, α and β at the equilibrium points. This analysis shows that for the augmented-state estimation problem there is no guarantee for exact recovery of α and β at equilibrium when using filtering techniques. The observability matrix derivation in this section is based on the linearized system around an equilibrium point. Therefore, it does not provide insight on parameter estimation for non-equilibrium trajectories, which we will instead explore numerically in the computational experiments in Section 4.4 using the PF described next.

4.4. **Estimation on synthetic data**

In this section, each parameter estimation routine described above is run on synthetically generated data. This is done to understand the potential to recover the true model parameters under controlled

settings. We show that all methods produce good estimates under non-equilibrium driving but have limited ability to recover true parameters under equilibrium driving conditions, consistent with the discussion from Section 4.3.

4.4.1. Setup of synthetic experiments

The general setup is as follows. First, synthetic data is created by selecting a set of model parameters and a predefined lead driver velocity profile. A time-series of velocity and space gap data is then created via a forward simulation of (4.5) under the selected parameters and input signal. The simulated data is then fed into each estimation method, with each returning a set of estimated parameter values. The accuracy of the recovered parameters and the resulting state error of the system trajectory under the recovered parameters is then compared. The run-time for each method is also reported.

Equilibrium driving

In order to create a set of synthetic measurement data for driving under equilibrium, we begin by setting the true parameters as: $\boldsymbol{\theta}_{true} = [0.08, 0.12, 1.5]^T$. These values are representative of parameter values that have been reported for commercial ACC systems [79]. Additionally, in order to generate a synthetic dataset, the velocity profile of a lead vehicle is needed, along with an initial space gap, and the initial velocity of the following vehicle.

We generate equilibrium data by setting the velocity of the lead vehicle at a constant $u_k = v_{lead} =$ 24 m/s while the initial space gap and the initial velocity of the follower satisfy $s_0 = \tau_{true} v_0$. A total of 900 seconds of velocity and space gap data is generated at a measurement frequency of 10 Hz. Because the data is generated such that the equilibrium condition (4.19) is satisfied, the true parameters α and β do not influence the space gap or velocity of the follower vehicle.

Non-equilibrium driving

We also consider a non-equilibrium driving scenario where the lead vehicle velocity (the system input) is empirically generated from a human driven vehicle in real highway driving. Using the empirical input, the ACC velocity and space gap data are still generated via forward simulation of the ACC model under the known true parameters. About 900 seconds of lead vehicle data is collected in which the driver of the vehicle follows the traffic rules but has variations in speed due interactions with other vehicles on the roadway. The simulation is initialized at a follower velocity and space gap of 32.5 m/s and 37.8 m, and the data is again generated at 10 Hz. Figure 4.1 displays the lead and follower velocity profiles, and the space gap data.

The true parameters used to generate the synthetic dataset are again $\boldsymbol{\theta}_{true} = [0.08, 0.12, 1.5]^T$, which is neither $\mathcal{L}_2$ (4.2) nor $\mathcal{L}_\infty$ (4.3) strict string stable. This means the ACC may amplify lead vehicle disturbances. As can be seen in the simulation data (Figure 4.1), in several occasions the follower vehicle slows down more than the leader.

4.4.2. Parameter estimation results on synthetic data

Using the synthetic data created above, we now turn to the results of each parameter estimation routine that attempt to recover the true parameters using only the measurement data. We use

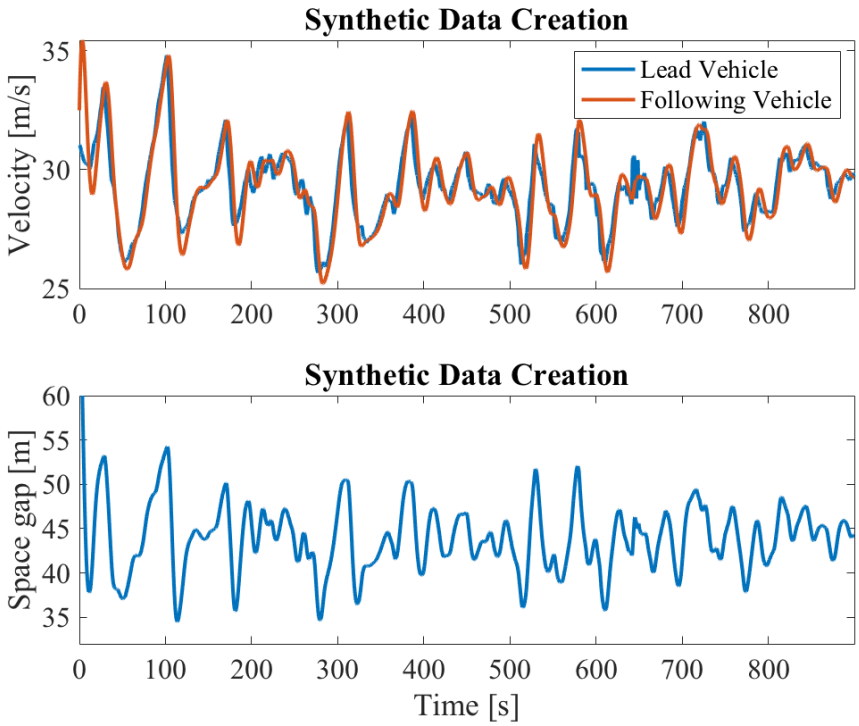


Figure 4.1: Synthetic space gap and following vehicle data, generated from an empirical lead vehicle profile.

Method	Parameters	Values
Batch optimization	$\boldsymbol{\theta}_0$	$[\mathcal{U}(0,1)\ \mathcal{U}(0,1)\ \mathcal{U}(1,3)]^T$
	# initial points	100
Least squares	$\boldsymbol{\gamma}_0$	$[0.976,\ 0.01,\ 0.01]^T$
	P_0	$0.1\boldsymbol{I}_3^{\ddagger}$
Particle filter	N_p	500 particles
	$\boldsymbol{\mu}_{x_0^a}$	$[37.8\ 32.5\ 0.1\ 0.1\ 1.4]^T$
	Q_0	$\text{diag}[0.5\ 0.5\ 0.2\ 0.2\ 0.3]^2$
	Q	$\text{diag}[0.2\ 0.1\ 0.01\ 0.01\ 0.01]^2$
	R	$\text{diag}[0.2\ 0.1]^2$

Table 4.1: Parameters and initialization for all estimation routines. ${}^{\ddagger}\boldsymbol{I}_3$ is the identity matrix with size 3×3.

the *mean absolute error* (MAE) in space gap and velocity to compare the accuracy of each estimation method. Both RLS and the PF require several algorithm parameters to be set, which are summarized in Table 4.1. For the batch method, we sample 100 starting points for the parameters from uniform distributions described in Table 4.1. For RLS, we set the initial coefficient vector $\boldsymbol{\gamma}_0$ and its corresponding covariance matrix P_0. For the PF, we set the number of particles used in the estimator N_p, the initial distribution of the augmented state vector (assumed to follow a normal distribution with mean $\boldsymbol{\mu}_{x_0^a}$ and covariance Q_0), and the model and measurement covariance matrices Q and R.

Equilibrium driving results

The performance of both online parameter estimators and the batch estimator are summarized in Table 4.2. The summary includes the estimated parameters, the corresponding MAE for space gap and velocity, and the run time.

Compared to the true parameters $\boldsymbol{\theta}_{true} = [0.08\ 0.12\ 1.5]^T$, all methods estimate the true τ accurately, but have larger errors on α and β. In the least squares method, this is due to the fact that the matrix X from (4.7) has rank 1 (i.e., the columns in X (4.8) are linearly dependent, since $v_i = v_j = u_i = u_j$ for $i, j \in \{1, ..., K\}$ at equilibrium), and consequently $\boldsymbol{\gamma}$ does not have a unique solution. In the PF, the lack of observability of the system (4.17) leads to the the non-convergence of the PDFs in the PF (see Figure 4.2). Only τ converges to the true value, while the distributions of α and β drift away from the true values over time. Unlike the RLS estimator and the PF, the batch method does not benefit from additional information about the true parameters via the prior given at time 0. As a consequence, the errors on α and β are largest for the batch method.

The experiments illustrate that for all methods, the parameters are not identified at equilibrium. This is consistent with the lack of persistent excitation in the input signal [27].

Even though the true α and β are not recovered correctly, all methods produce models that have negligible MAE for speed and space gap. This is again due to the fact that α and β do not influence the trajectory of the ACC vehicle at equilibrium.

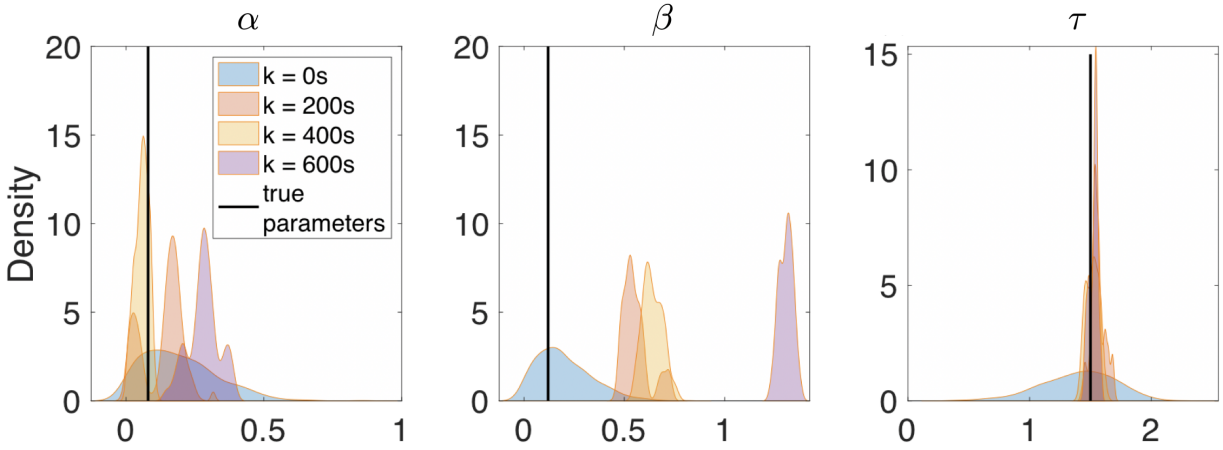


Figure 4.2: Posterior parameter PDFs for equilibrium driving dataset from PF estimates. The plot shows that parameter α and β are not identified correctly, i.e., the distributions drift away from the true values (black vertical lines) over time. Only the distribution of τ converges to the true value.

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 8.34$ $\beta = 7.30$ $\tau = 1.50$	$\alpha = 0.0965$ $\beta = 0.0976$ $\tau = 1.50$	$\alpha = 0.065$ $\beta = 0.604$ $\tau = 1.50$
Algorithm	Offline	Online	Online
Running time (s)	12.44	0.06	8.45
MAE space gap (m)	0.00	0.00	0.14
MAE velocity (m/s)	0.00	0.00	0.00

Table 4.2: Performance on synthetic equilibrium data. True parameters are: $\alpha_{true} = 0.08, \beta_{true} = 0.12, \tau_{true} = 1.5$.

Finally, we compare the runtime of each method. The recursive least-squares approach recovers the parameters in 0.06 seconds, which is the fastest runtime of the three methods by more than 2 orders of magnitude. The PF runs in 8 seconds, which is a factor of 100 faster than real time (recall the dataset is 900 seconds long). Although the total time to execute the batch method on a 900-second data is only 12 seconds, this is an offline method and cannot be run until all the data has been collected. This is in contrast to the online methods, which produce new estimates every time a new measurement is available. All the experiments are performed on the same MacBook Pro with 2.7 GHz CPU, such that the running time is comparable.

Non-equilibrium driving results

We now turn to the performance of the methods on the synthetic dataset generated from the ACC model when fed empirically collected non-equilibrium lead vehicle driving data. The results of each method are summarized in Table 4.3.

Both the batch method and the RLS estimator recover the true model parameters used to generate the data. The PF performs worse but the estimates improve over time (Figure 4.3). When simulating with the mean values of the PF parameter estimates, the calibrated model has an MAE of 0.32 m/s in velocity and 2.54 m in the space gap. Given that the PF assumes both a model and measurement noise, it is not surprising that it does not perfectly recover the true parameters (since the model and measurement in fact have zero error in this synthetic example). We note the runtime for the two online methods again outperforms the batch method, and are significantly faster than real time.

Table 4.3 also shows the string stability estimate based on the recovered parameters using each method. Because the value of the parameters are used to determine the model string stability, it is important to know if errors in the estimates are large enough to change the string stability estimate. In this experiment, we find that all models under the estimated parameters are string unstable, as is the case with the true model parameters.

Additional numerical experiments can be found in [243], which explores the RLS performance in the presence of measurement noise, looking both at the real noise expected from the stock sensors as well as the sensitivity of the estimator to a range of noises. We find that the method can tolerate the noise levels present on the commercial vehicle platforms.

4.5. Case study on a 2019 ACC equipped vehicle

We now present a case study in which all methods are used to estimate the model parameters using data collected from a 2019 ACC equipped vehicle.

4.5.1. Experimental details

We first describe the velocity and space gap data collection using measurements from the stock radar system of an ACC-equipped vehicle. We compare the measurements collected from the ACC vehicle CAN bus to measurements collected using those from the GPS devices mounted on the ACC vehicle and its leader during the experimental data collection.

To set up the experiment, we create a two-vehicle system in which an ACC-equipped vehicle follows an instrumented lead vehicle driven by a human in real freeway driving conditions. The vehicle used in this experiment is a commercially available 2019 SUV with a full velocity range adaptive

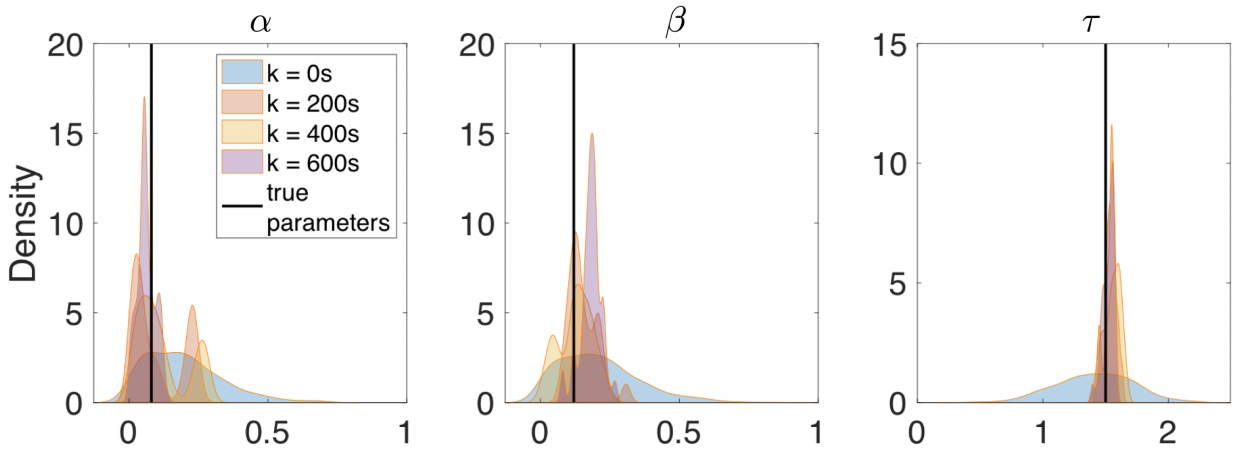


Figure 4.3: Posterior parameter PDFs for non-equilibrium driving dataset from PF estimates.

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 0.08$ $\beta = 0.12$ $\tau = 1.5$	$\alpha = 0.08$ $\beta = 0.12$ $\tau = 1.5$	$\alpha = 0.04$ $\beta = 0.21$ $\tau = 1.41$
Algorithm	Offline	Online	Online
Running time (s)	11.27	0.06	8.43
MAE space gap (m)	0.00	0.00	2.54
MAE velocity (m/s)	0.00	0.00	0.32
$\mathcal{L}_2$ strict string stable	No	No	No
$\mathcal{L}_\infty$ strict string stable	No	No	No

Table 4.3: Performance on synthetic nonequilibrium data: True parameters are: $\alpha_{true} = 0.08, \beta_{true} = 0.12, \tau_{true} = 1.5$. The reported parameter values are the maximum a posteriori (MAP) estimates of the last timestep (at 900 s), which can be slightly different from the estimates of the earlier timesteps.

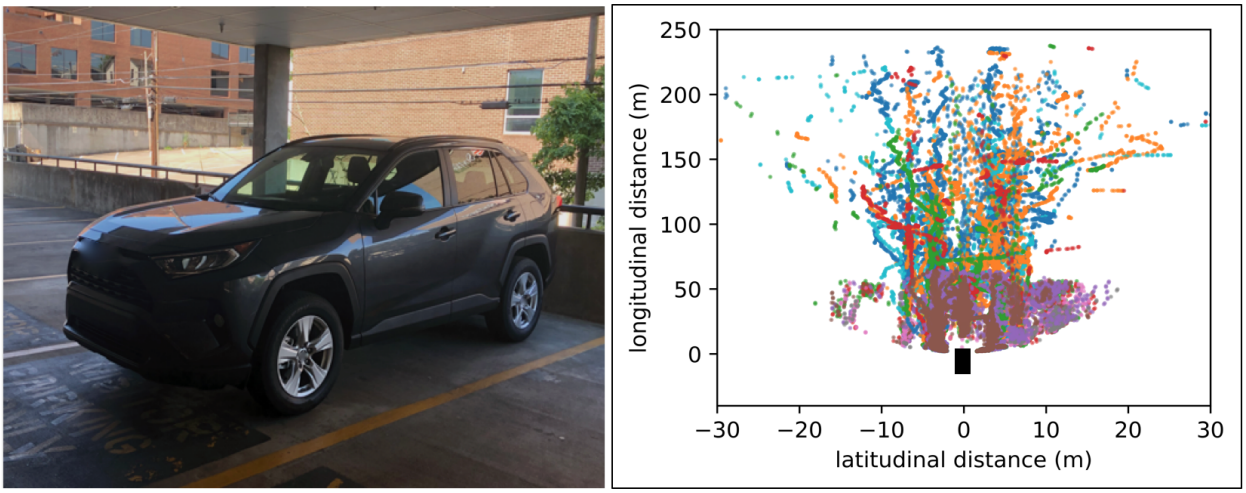


Figure 4.4: *Left*: The 2019 ACC equipped stock SUV used in the experiment. *Right*: relative position data collected from the CAN bus of the vehicle within a duration of 15 min drive. Each point corresponds to the latitudinal and longitudinal distance to an object detected by the stock radar sensor and reported on the CAN bus. Colors correspond to distinct objects.

cruise control system (Fig. 4.4, left). A total of 15 minutes (900 seconds) of data are recorded at 10 Hz in which the ACC vehicle follows the lead vehicle through traffic on a freeway in Nashville, TN.

The driver of the lead vehicle is instructed to drive as they would normally in traffic, while the ACC equipped vehicle follows with ACC engaged. The entire experiment is conducted without any ACC de-activations or overrides, and no cars cut in between the leader and the ACC follower.

Velocity, lead vehicle velocity, and space gap data is collected by recording measurements from the CAN bus on the vehicle. The radar unit reports the latitudinal and longitudinal distance to objects in front of the vehicle (Figure 4.4, right), from which the space gap between the two vehicles can be computed. It also reports the relative velocity of objects in the field of view of the sensor. Since the ACC vehicle velocity is also published to the CAN bus, the lead vehicle velocity can be determined from the radar data. With straightforward processing of the radar data, it is possible to convert into velocity data of the leader and follower, and space gap, as shown in Figure 4.5.

4.5.2. **CAN bus velocity and space gap data validation**

In order to assess the accuracy of the stock vehicle radar unit, both vehicles are additionally equipped with sub-meter accurate GPS units which track global position and velocity. The devices are the same units used for primary data collection in our previous work [79, 80, 192]. The time-series of space gap and velocity are recorded from both GPS devices and the radar unit on the ACC vehicle and compared.

A histogram of the differences between the two measurement techniques is displayed in Figure 4.6. The distribution of differences between radar space gap measurements and GPS space gap measurements is approximately zero-centered, as are the relative velocity differences. The sensors do not appear to be biased. The standard deviations of the differences between the measurement devices

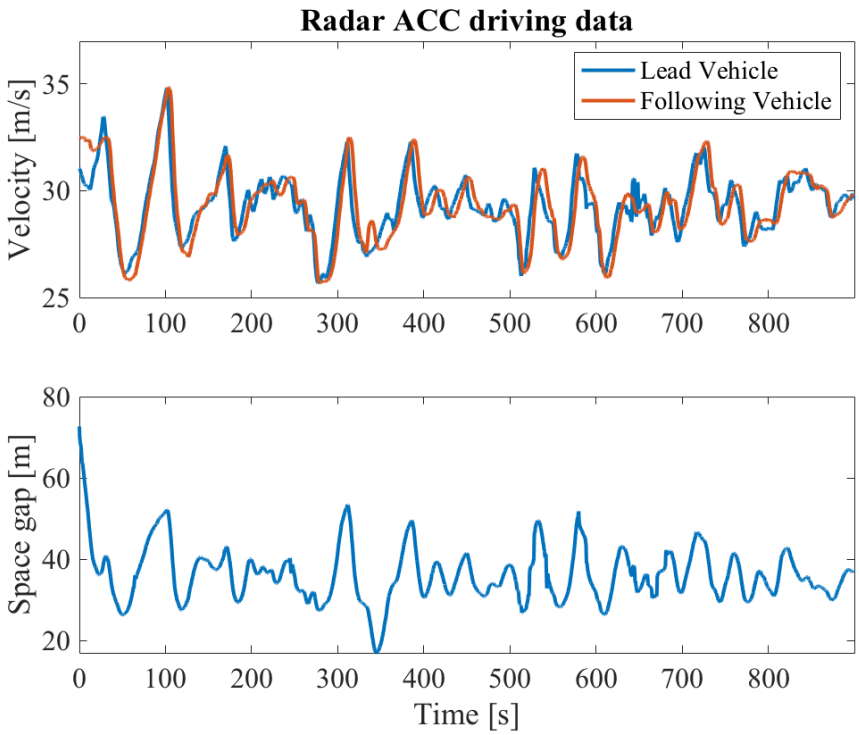


Figure 4.5: CAN bus measurement of an ACC-enabled vehicle following a human-driven vehicle.

is 0.72 m and 0.20 m/s for the space gap and relative velocity respectively, suggesting the sensor noises are also low.

As a further check, we briefly note that all estimation methods described next were run on data collected from the GPS devices as well as on the data from the radar unit as logged on the CAN bus. In all cases the estimated parameters are similar across the two sensor platforms. We conclude that the on-board radar measurements reported on the CAN bus are comparable to the GPS devices.

The fact that the space gap, velocity, and relative velocity data can be collected directly from the CAN bus significantly simplifies experimental data collection. Compared to our earlier work that required instrumenting two vehicles, the approach here can be applied using only a single vehicle (the ACC equipped vehicle). Eventually this may allow improved data collection from ACC vehicles in increasingly realistic settings, such as under cut-ins and lane changing.

4.5.3. **Parameter estimation results on a 2019 ACC vehicle**

With the CAN bus data validated, we now turn to the parameter estimation problem applied to the vehicle. The data contains non-equilibrium driving data, which is used to estimate the parameters using each method. We follow the same setup as the synthetic data experiments, with the notable exception that the true parameters of the ACC vehicle are unknown. The MAE between the measured space gap and the space gap under the estimated parameters is reported. Similarly, the MAE of the velocity is used to assess the quality of the estimated parameters. The string stability of the calibrated model under the parameters estimated by each method is also determined.

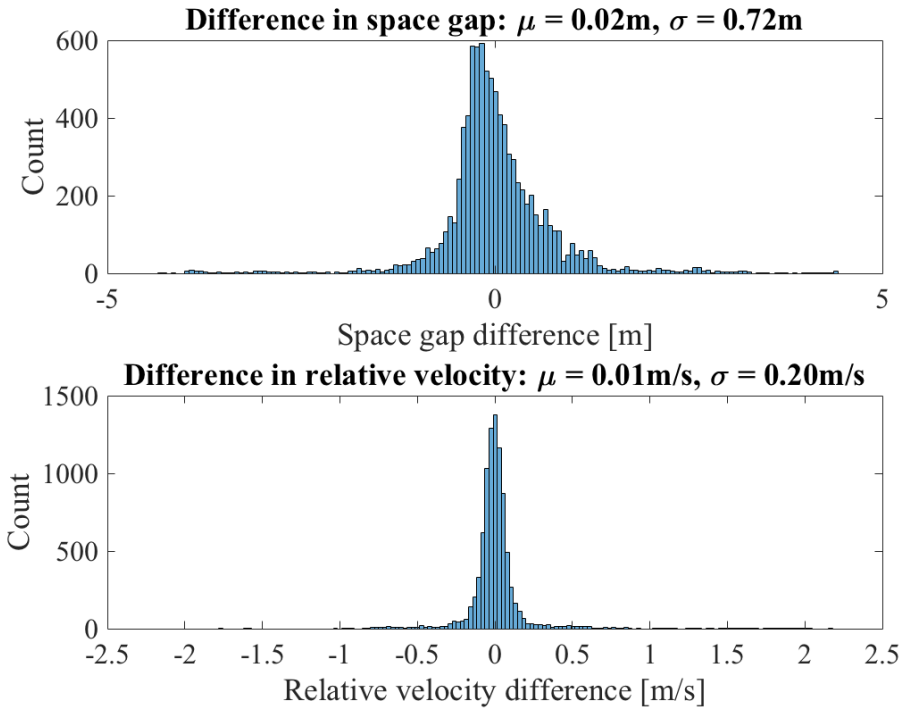


Figure 4.6: Histogram of the difference between GPS measurements and CAN bus measurements for space gap and relative velocity measurements.

Criteria	Batch optimization	RLS	PF
Estimated parameter values	$\alpha = 0.0227$ $\beta = 0.194$ $\tau = 1.227$	$\alpha = 0.0174$ $\beta = 0.164$ $\tau = 1.127$	$\alpha = 0.0431$ $\beta = 0.164$ $\tau = 1.221$
Algorithm	Offline	Online	Online
Running time (s)	11.98	0.06	8.70
MAE space gap (m)	2.02	2.24	2.60
MAE velocity (m/s)	0.24	0.26	0.35
$\mathcal{L}_2$ strict string stable	No	No	No
$\mathcal{L}_\infty$ strict string stable	No	No	No

Table 4.4: Performance summary of all estimation methods on ACC data.

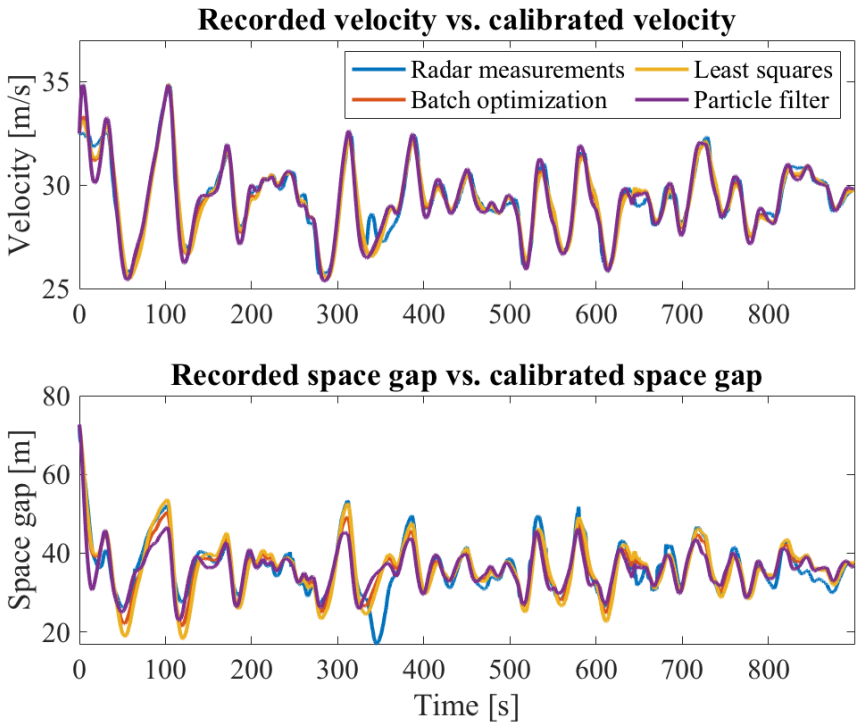


Figure 4.7: Comparison between recorded vehicle velocity and space gap vs simulated for each model found.

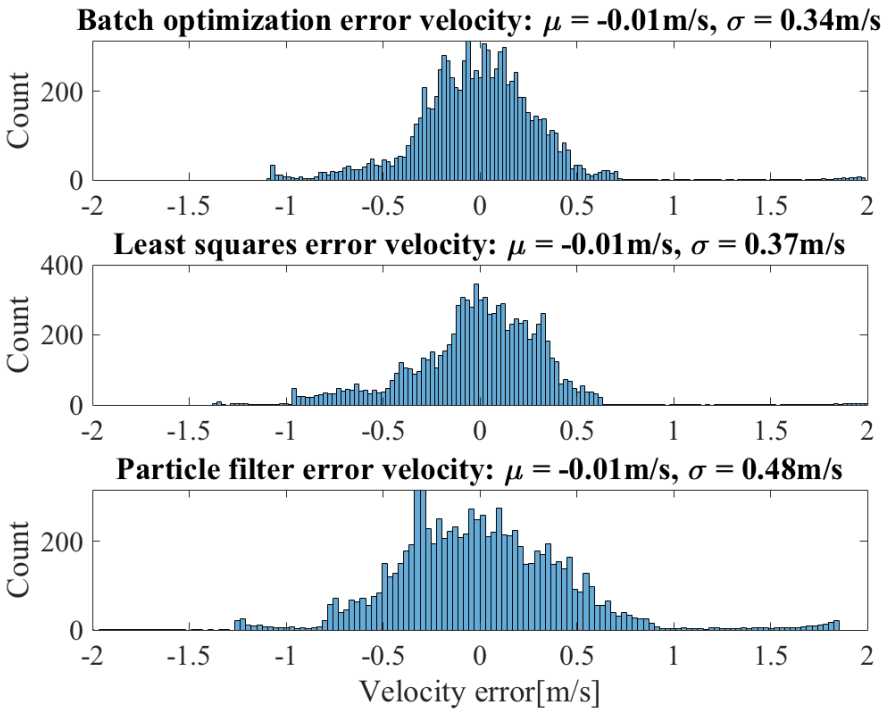


Figure 4.8: velocity error distribution for each calibrated model.

These results are summarized in Table 4.4. All methods produce parameters that fit the data well, with some differences in the actual parameter values. A simulation using the estimated parameters from each method is shown in Figure 4.7. All methods have nearly identical velocity profiles, with slight differences in the space gap profiles. The batch optimization achieves both the lowest MAE velocity and space gap errors at 0.24 m/s and 2.02 m. This represents errors of 0.8% in velocity and 5.0% in space gap. The least-squares method has a comparable performance, with MAE values of 0.26 m/s and 2.24 m (0.87% in velocity and 5.6% in space gap). Finally, the PF estimated parameters produce slightly higher MAEs of 0.35 m/s and 2.60 m, which correspond to percent errors of 1.2% in velocity and 6.5% in space gap. Overall, the MAEs are comparable and low both in absolute values and in percent. Moreover, the models are similar in scale to those found in other works [146, 79].

We explore the errors in more detail. The largest error between the measured data and the ACC model run with estimated parameters occurs between roughly 325 seconds and and 375 seconds, in which the real ACC vehicle engages in an acceleration that is not captured by any of the calibrated models. This underscores that while each calibrated model produces a good overall reconstruction of the ACC vehicle, none are able to perfectly describe the complex nonlinear vehicle dynamics controlled by a proprietary ACC system.

The histograms of the errors are shown in Figure 4.8 and 4.9. The average velocity and space gap error of both online methods is relatively similar to that of the batch method. All three methods return a model that has an average velocity error of within 0.01 m/s and similar standard deviations.

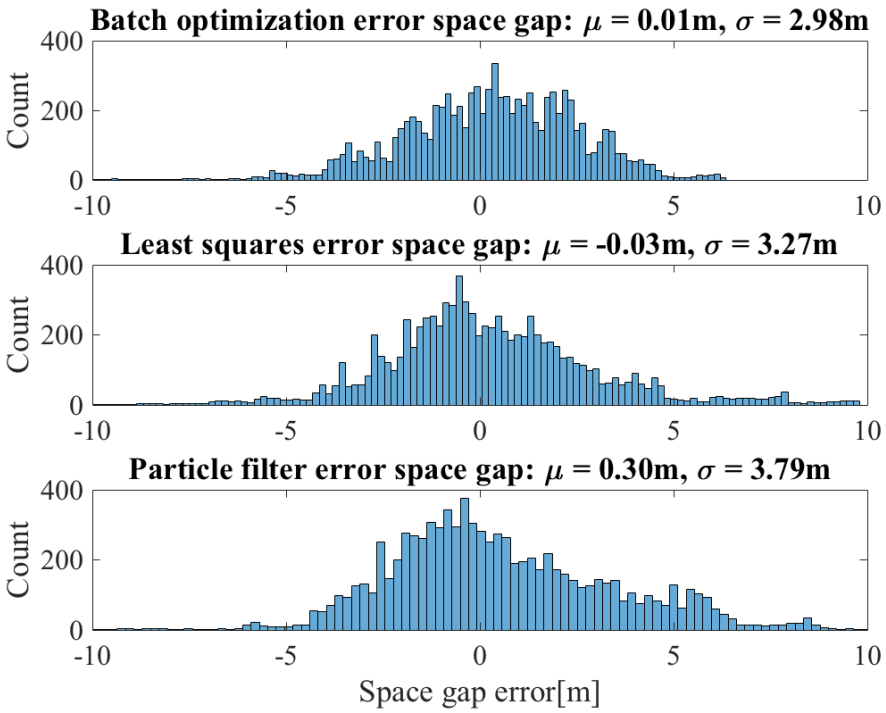


Figure 4.9: space gap error distribution for each calibrated model.

Both the online methods produce the average space gap errors that are slightly more biased than the batch method, and standard deviations within 4 m. These MAE values are of the same order as is reported in other works [146, 79, 80]. Given that both the online methods have similar average velocity and space gap errors compared to the batch methods and the estimated models fit the recorded data relatively well, this suggests that both the RLS and the PF are viable online methods for learning ACC model parameters.

With the estimated parameters from each of the methods, we check the string stability of the vehicle under ACC control. Like all previous studies considering commercial ACC systems [146, 79, 80, 110], we find that the calibrated model of the vehicle tested in this work is neither $\mathcal{L}_2$ or $\mathcal{L}_\infty$ strict string stable. The results are consistent across the different calibration methods, as summarized in Table 4.4.

With respect to the runtime, the recursive least-squares method is again the fastest, with a total computation time of 0.06 seconds to process the 15 minute dataset. The PF executes in 8.7 seconds, while the batch optimization method runs in 11.98 seconds. The runtime of the batch method is sensitive to the initial guess and the number of parameters to be estimated. The online methods have a distinct advantage in real-time applications, since they produce new estimates of the parameters as new data becomes incrementally available, and they can scale to arbitrarily long datasets.

4.6. **Conclusion**

This work uses two online methods to estimate parameters of vehicles under control of a stock ACC system, and provides a corresponding parameter identifiability analysis for the estimators. The online methods used here are scalable and suitable for real time implementations, and produce comparable results to an offline batch optimization method. All methods are tested on a 2019 vehicle with ACC using sensor data from the stock vehicle platform as reported on the CAN bus. All methods indicate the vehicle ACC system is string unstable, adding to the findings of eight other ACC systems as reported in [79, 80]. We further intend to exploit the experimental platform used in this work, which allows critical velocity and space gap data to be recorded directly from the CAN bus of the vehicle. Such data could be valuable for a variety of experimental settings in which humans and automation systems interact in mixed traffic.

We envision in our own work to generalize the application of the online system identification methods to study human driving behavior, using vehicles equipped with stock sensors similar to the ACC vehicle used in this work. As humans or automated vehicles may change driving behaviors depending on traffic conditions or environmental factors, it is important to apply online system identification algorithms to capture such behavioral changes characterized by the model parameters.

CHAPTER 5

Applications on human car-following system identification

5.1. Introduction

As discussed in Chapter 4, ACC automatically adjusts the longitudinal speed of the ego vehicle to maintain a safe distance from the vehicle ahead. ACC has been shown to increase safety, enhance driving comfort, and reduce fuel consumption [187, 174, 193, 249]. However, the limited settings of ACC prevent the drivers to preserve their own car-following styles, resulting in lack of trust and usage of that technology. In addition, a variety of usage conditions and the changing of the drivers' expectations persist in real-world driving. Drivers differ in their preferences and skills, and their styles may change over time. Personalization on vehicle maneuvers such as path tracking, steering and car-following is less developed, yet implicit driving preference significantly impacts driver's acceptance and trust towards the existing advanced driver-assistance systems (ADAS) [84]. In this chapter we formulate the penalization task as a system identification problem, where we design a personalized adaptive cruise control (PACC) that is learned from human-driving data.

The learning-based model benefits from the exploratory data-driven tools, as opposed to the model-based system identification which are often based on a fixed model structure. However, challenges still persist such as the verification of safety, stability and rationality. Recent developments such as the control Lyapunov and control barrier functions have been applied to provide safe and stable controlled systems [5, 6], as well as formal verification tools to facilitate assured autonomy from learning [258, 225, 4].

Amongst all the control design approaches, we draw particular attention to Gaussian Process (GP) regression, to design a personalized ACC that mimics the individual's driving behaviors. We focus on predicting the drivers' preferences and actions based on their past behaviors. GP regression can be utilized to identify the relationship between input (driver's perceived information) and output (desired acceleration), which allows it to provide personalized guidance towards driving.

In this chapter we make the following contributions:

- **Design of a GP-based personalized adaptive cruise controller (GP-PACC)** that allows learning the implicit longitudinal human driving styles without categorizing it based on predefined rules. This approach is purely data-driven, and allows each user to have a unique hyperparameter set that characterize his/her driving profile.
- **Validation of the proposed GP-PACC with both the synthetic data and human-in-the-loop experiments**. Results show that GP-PACC can almost exactly recovers the synthetic car-following data even under reasonable measurement noises, and can capture the driving styles of a real human driver up to 80% more accurately than calibrated baseline car-following models.

The remainder of this chapter is organized as follows: Section 5.2 introduces the problem formulation of this study. Section 5.3 outlines the fundamentals of GP regression used to model car-following behaviors, and describes the training and validation method for the GP model. In section 5.4 we conduct numerical experiments and human-driving experiments on a game engine to test the validity of the model. Finally, the study is concluded with some future directions in section 5.5.

5.2. **Problem formulation**

The proposed GP model aims to take the same input-output configuration of a traditional ODE-based car-following model (3.2). It learns from individual driver's naturalistic car-following behavior, and outputs a desired acceleration profile that closely mimics it.

Consider the discretized car-following dynamics of the general continuous-time form (3.2)

$$x_{k+1} = \begin{bmatrix} s \\ v \end{bmatrix}_{k+1} = \begin{bmatrix} s_k + (u_k - v_k)\Delta t \\ v_k + f_{\mathrm{CF}}(s_k, v_k, u_k)\Delta t \end{bmatrix} \tag{5.1}$$

where $f_{\mathrm{CF}} : \mathbb{R}^3 \to \mathbb{R}^1$ is a GP model. GP is trained to recover driver's naturalistic car-following behavior by minimizing the difference between the predicted acceleration and the recorded naturalistic driving acceleration.

The general assumption of GP regression is that inputs and outputs follow a multivariate Gaussian distribution, where any collection of the input/output vectors are jointly Gaussian distributed. GP regression is nonparametric (i.e. not limited by a functional form), so rather than calculating the probability distribution of parameters of a specific function, GP calculates the probability distribution over all admissible functions that fit the data. The GP training requires us to specify a prior distribution, and maximize the posterior probability of hyperparameters (that describe the prior) using the training data. The trained hyperparameters can then be used to make prediction on any points of interest. The general form of the prior is:

$$f_{\mathrm{GP}}(\mathbf{z}) \sim \mathcal{GP}(m(\mathbf{z}), c(\mathbf{z}, \mathbf{z}^{'})), \tag{5.2}$$

where $m(\mathbf{z})$ is the mean function of the input vector $\mathbf{z}$, and $c(\mathbf{z}, \mathbf{z}^{'})$ is a covariance function, which encodes the smoothness of f_{GP} (i.e., similarity of inputs corresponds to the similarity of outputs). Furthermore, we adopt a nonlinear output error (NOE) training procedure outlined in [111], with the goal of making GP a good simulation model to achieve multiple-step-ahead prediction (simulation) accuracy, instead of just one-step-ahead prediction accuracy.

The block diagram of the proposed GP-PACC system is shown in Fig. 5.1. We consider the ACC algorithm as the high-level controller, which takes the input of the ego vehicle speed, lead vehicle speed, and space gap information, and outputs an acceleration. The low-level vehicle dynamics will then output the corresponding speed and space gap.

5.3. **System identification via Gaussian Processes**

In this section we outline GP training using the nonlinear output error (NOE) approach. The derivation is based on [112]. The goal of NOE is to include simulation in the training step, thus reducing the GP multi-step prediction (simulation) error, as apposed to naive GP regression that is trained on one-step prediction only.

Gaussian processes extend multivariate Gaussian distributions to infinite dimensionality. They are a form of supervised learning and the training result represents a nonlinear mapping $f_{\mathrm{GP}}(\mathbf{z}) : \mathbb{R}^{\dim(\mathbf{z})} \to \mathbb{R}$, such as f_{CF} in (3.2). The mapping between the input vector $\mathbf{z}$ and the function value $f_{\mathrm{GP}}(\mathbf{z})$ is accomplished by the assumption that $f_{\mathrm{GP}}(\mathbf{z})$ is a random variable and is jointly Gaussian distributed with $\mathbf{z}$, which is also assumed to be a random variable [176].

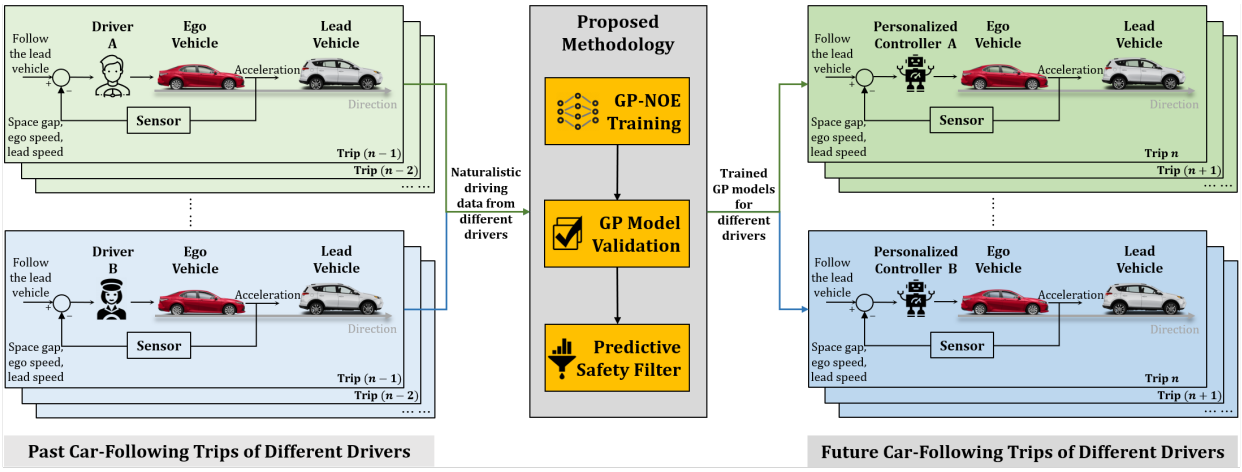


Figure 5.1: Block diagram of the proposed GP-PACC system. The predictive safety filter component is described in Appendix B

Setup The GP model setup includes selecting the model regressors, the mean function and the covariance function. In the following discussion, we focus on the commonly used zero-mean and the squared-exponential covariance function that relates two sample input vectors $\mathbf{z}_i$ and $\mathbf{z}_j$:

$$c(\mathbf{z}_i, \mathbf{z}_j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{z}_i - \mathbf{z}_j)^T P^{-1}(\mathbf{z}_i - \mathbf{z}_j)\right) + \sigma_n^2 \delta_{ij}, \tag{5.3}$$

where $\delta_{ij} = 1$ if $i = j$ and $\delta_{ij} = 0$ otherwise, and $P = \text{diag}[l_1^2, ..., l_{\dim(\mathbf{z})}^2]$ contains the characteristic length scale for each dimension of the input vector. The hyperparameters of the covariance function $\theta = [\sigma_f, \sigma_n, l_1, ..., l_{\dim(\mathbf{z})}]^T$ include the measurement noise σ_n, the process standard deviation σ_f, and the characteristic length scales, which are learned by maximizing the likelihood of the observation.

Bayesian model inference The inference of a Bayesian model is a process where the prior knowledge of the hyperparameter vector θ is updated to a posterior distribution through the identification (training) data.

We specify the training input $\mathbf{Z}$ and target $\mathbf{y}$ for a total of N samples:

$$\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_N]^T \tag{5.4}$$

$$\mathbf{y} = [y_1, y_2, ..., y_N]^T, \tag{5.5}$$

where the subscript denotes the sample index.

The corresponding GP model can be used for predicting the function value y_* given a new input $\mathbf{z}_*$ based on a set of past observations $\mathcal{D} = \{\mathbf{Z}, \mathbf{y}\}$. The key assumption is that the data can be represented as a sample from a multivariate Gaussian distribution:

$$\begin{bmatrix} \mathbf{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K & K_*^T \\ K_* & K_{**} \end{bmatrix}\right), \tag{5.6}$$

where $\mathbf{0} \in \mathbb{R}^N$ is a vector of zeros, and K is the covariance matrix

$$K = \begin{bmatrix} c(\mathbf{z}_1, \mathbf{z}_1), c(\mathbf{z}_1, \mathbf{z}_2)...c(\mathbf{z}_1, \mathbf{z}_N) \\ c(\mathbf{z}_2, \mathbf{z}_1), c(\mathbf{z}_2, \mathbf{z}_2)...c(\mathbf{z}_2, \mathbf{z}_N) \\ ..., ... \\ c(\mathbf{z}_N, \mathbf{z}_1), c(\mathbf{z}_N, \mathbf{z}_2)...c(\mathbf{z}_N, \mathbf{z}_N) \end{bmatrix} \tag{5.7}$$

$$K_* = [c(\mathbf{z}_*, \mathbf{z}_1), c(\mathbf{z}_*, \mathbf{z}_2)...c(\mathbf{z}_*, \mathbf{z}_N)] \quad K_{**} = c(\mathbf{z}_*, \mathbf{z}_*). \tag{5.8}$$

We want to infer θ by computing the posterior distribution of the hyperparameters:

$$p(\theta|\mathbf{Z}, \mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{Z}, \theta)p(\theta)}{p(\mathbf{y}|\mathbf{Z})}. \tag{5.9}$$

For unknown knowledge of θ, it is reasonable to specify a uniform distribution $p(\theta)$, and as a result, the posterior distribution is proportional to the marginal likelihood, i.e.,

$$p(\theta|\mathbf{Z}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{Z}, \theta). \tag{5.10}$$

Maximizing the posterior distribution is equivalent to minimizing the negative log likelihood $l(\theta)$:

$$l(\theta) := \ln\, p(\mathbf{y}|\mathbf{Z}, \theta) = -\frac{1}{2}\ln|K| - \frac{1}{2}\mathbf{y}^T K^{-1}\mathbf{y} - \frac{N}{2}\ln(2\pi). \tag{5.11}$$

Once the best-fit θ is obtained, we can compute the covariance matrix (5.7) and the output distribution y_* (in terms of the prediction mean and variance) given a new input vector $\mathbf{z}_*$:

$$\begin{aligned} \hat{y}_* &= K_* K^{-1}\mathbf{y} \\ \text{var}(y_*) &= K_{**} - K_* K^{-1} K_*^T. \end{aligned} \tag{5.12}$$

For the simplicity of notation, we denote the output prediction as:

$$y_* = f_{GP}(\mathbf{z}_*, \theta) + \mathcal{N}(0, \sigma_n^2). \tag{5.13}$$

We adopt a training process similar to calibrating an ODE-based car-following model [133, 233, 41, 166, 81]. The process is to find the parameters of which the simulated output is closest to the recorded measurement. The simulated state $\{\hat{x}_k = [\hat{s}, \hat{v}]_k\}_{k=1}^N$ given the initial state $x_0 = [s_0, v_0]$, the external input signal $u_{0:N-1}$ and a GP model (5.13) are used as part of the pseudo training input of the GP-NOE model. This way, the dynamics (3.2) can be inherently included in the training when the simulated states are fed back as the regressors. The simulated state can be obtained via:

$$\begin{aligned} \hat{x}_{k+1} = \begin{bmatrix} \hat{s} \\ \hat{v} \end{bmatrix}_{k+1} &= \begin{bmatrix} \hat{s}_k + (u_k - \hat{v}_k)\Delta t \\ \hat{v}_k + f_{GP}(\hat{\mathbf{z}}_k, \theta)\Delta t \end{bmatrix} \\ \hat{x}_0 = x_0 = [s_0, v_0], k &= 0 : N-1 \end{aligned} \tag{5.14}$$

where $\hat{\mathbf{z}}_k = [\hat{s}_k, \hat{v}_k, u_k]$ is the k^{th} sample of the pseudo training input, which contains the simulated state and the measured external input at time k, as opposed to the recorded data $\mathbf{z}_k = [s_k, v_k, u_k]$. We rewrite (5.14) to the following to denote the simulated trajectories:

$$\hat{x}_{k+1} = g(\hat{x}_k, u_k, \theta, \Delta t), k = 0 : N-1. \tag{5.15}$$

The simulation also requires an initial guess of the hyperparameters θ. The mean prediction is stated as $f_{GP}(\hat{\mathbf{z}}_k, \theta)$ according to (5.12). The training target is the acceleration data at the same timestep $\mathbf{y}_{1:N}$.

Let us denote $\hat{\mathbf{Z}}_{1:N} = [\hat{\mathbf{z}}_1, \hat{\mathbf{z}}_2, ..., \hat{\mathbf{z}}_N]^T$. The training of the GP model with NOE structure is an iterative process shown in **Algorithm 2**. The implementation is based on the GP-Model-based System-Identification Toolbox for Matlab [191].

Algorithm 2 GP-NOE training

Data: Training input $\mathbf{Z}$, training target $\mathbf{Y}$, covariance function $c(\cdot, \cdot)$, initial hyperparameters θ, initial condition $x_0 = [s_0, v_0]$

while $l(\theta)$ (5.11) *is not minimal* **do**

  obtain the simulated (pseudo) regression vectors $\hat{\mathbf{Z}}_{1:N}$ with the initial state $x_0 = [s_0, v_0]$ and the current hyperparameters θ, according to (5.14);

  update θ by minimizing the negative log likelihood $l(\theta)$.

end

5.3.1. Model validation

Training a GP-NOE model is similar to calibrating a car-following model, which is conducted by finding the model parameters that minimize the error between the *simulated* vehicle trajectories and the benchmark. We validate the GP model in simulation, i.e., obtaining a closed-loop simulated trajectory according to (5.14), and compare the acceleration and space-gap trajectories with the recorded data, similar to evaluating a car-following model from calibration (e.g., [133, 233, 41, 166, 81, 48]).

Two performance metrics are measured: the mean squared error (MSE) and the log predictive-density error (LPD) [112, 72] between the GP simulated acceleration and the recorded acceleration of a validation data set:

$$\begin{aligned} MSE &= \frac{1}{N}\sum_{k=1}^{N}(y_k - \hat{y}_k)^2 \\ LPD &= \frac{1}{2}\ln(2\pi) + \frac{1}{2N}\sum_{k=1}^{N}\left(\ln(\sigma_k^2) + \frac{(y_k - \hat{y}_k)^2}{\sigma_k^2}\right) \end{aligned} \tag{5.16}$$

where y_k is the acceleration data at timestep k, $\hat{y}_k$ is the mean prediction of GP at timestep k, and σ_k^2 is the prediction variance. MSE measures the error only on the mean predicted acceleration, whereas LPD takes into account the entire distribution of the prediction by penalizing the overconfident prediction (smaller variance) more than the acknowledged bad predicting values (higher variance). In addition, the MSE on the space gap (MSE-s) will also be calculated, since small and biased acceleration prediction might lead to a larger space gap error. The simulated space gap can be obtained from the GP output using (5.14). The lower these measures the better GP model performs in terms of recovering the original driving data.

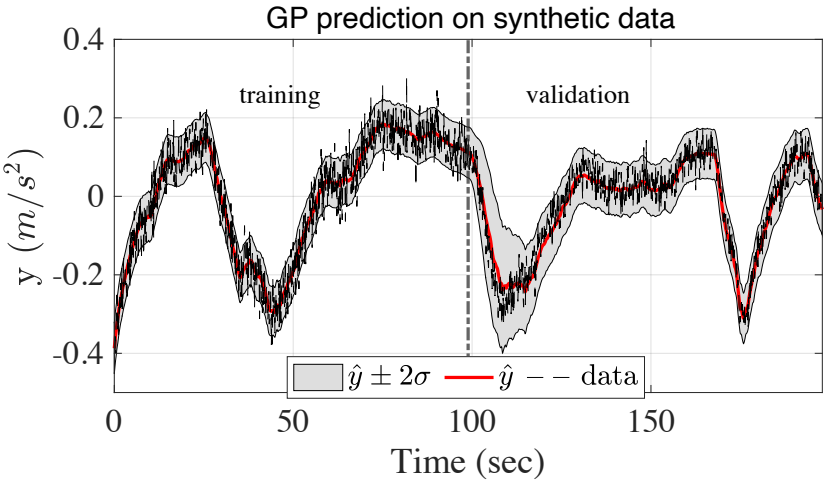


Figure 5.2: Compare GP predicted acceleration (red solid line) with data (black dotted line). The first half is training result and the second half is validation result.

5.4. **Experiments and results**

5.4.1. **Numerical experiments**

A set of car-following data is synthetically generated using IDM [208], which has been used throughout the literature to model a realistic driver behavior, such as asymmetric accelerations and decelerations. The simulated car-following data serves as "ground truth", from which the GP-PACC prediction errors can be computed, and the prediction variance can be compared with the known noise. The model is specified in (3.10). In this experiment, the synthetic data is obtained from an IDM with parameters $\theta = [s_j, v_f, T, a, b, \delta] = [2, 33.3, 1.6, 0.73, 1.67, 4]$ based on empirical investigations [208].

We generate 200 seconds of data at 10Hz given a pre-recorded, freeway high-speed lead vehicle speed profile ranging between 25m/s to 35m/s. The simulated data is also manually polluted with Gaussian white noise ranging from 0.01 to 0.1 standard deviation onto the acceleration signal, in order to emulate the realistic sensor errors. We train the GP model on the first 100 seconds and use the second half as the validation set (see Fig. 5.2). This composition is shown to reproduce the car-following styles for various drivers consistently well in our later experiments.

Fig. 5.2 visualizes the GP simulated acceleration (red solid line) and the benchmark data (black dashed line), as well as the prediction uncertainty (grey area). The data is synthetically generated using IDM. One can see that the uncertainty band well captures the deviation of the data set, and the mean prediction traces the mean of the data accurately.

More quantitatively, Fig. 5.3 shows the MSE of the GP simulation on the acceleration, velocity and the space gap, as well as the LPD on the acceleration, respectively. When various levels of sensor noises are present, the GP results show that the MSE of acceleration prediction is overall very low (under 3.5×10^{-4}), and so does the corresponding velocity (under $0.01(m/s)^2$) and space gap MSE

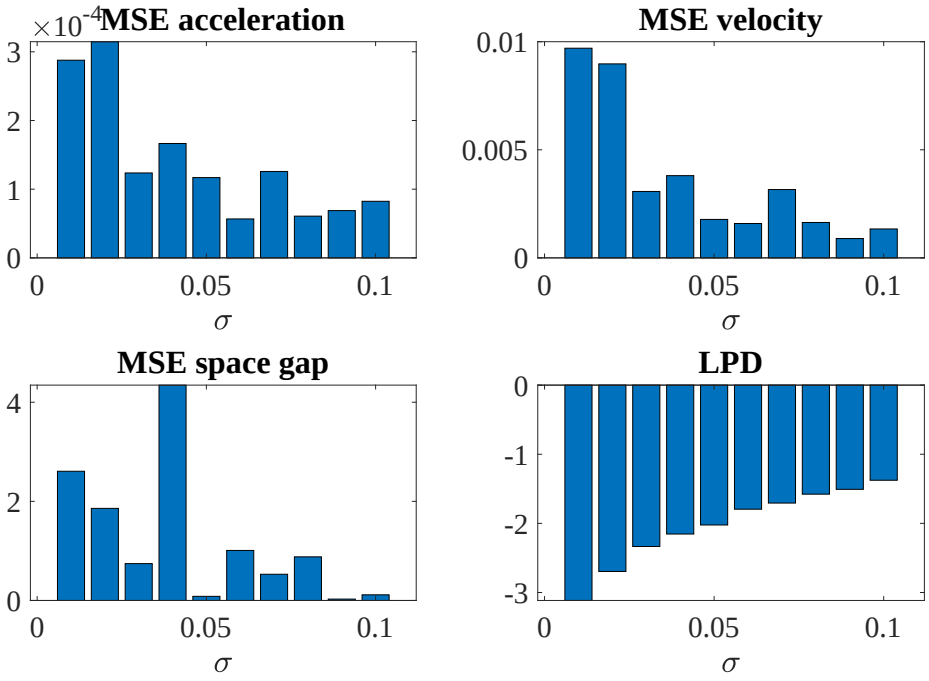


Figure 5.3: Performance of GP-PACC compared with synthetic data.

(under $4.5m^2$). It indicates that the GP model can very accurately reproduce the driving profile and is robust under noisy measurements.

Note in Fig. 5.3 that, as the standard deviation of the added noise increases (emulating a higher noise of real-world acceleration measurement), the MSE values for both the acceleration and the space gap prediction are lower. There are two reasons for this: (a) inverting the covariance matrix K during the parameter inference step (5.11) suffers from numerical issues when the variance of $\mathbf{y}$ is too low; (b) Training may not converge to a global minimal due to the non-convex and non-smooth objective function (5.11), albeit the warm start.

Lastly, the LPD (bottom of Fig. 5.3) on the acceleration prediction indicates that the new observations (from the validating set) are well-accounted by the posterior predictive distribution, even with higher sensor errors.

Overall, the numerical experiments suggest that GP can accurately reproduce the driving data even with reasonable measurement noise. The posterior distribution can also accurately characterise the uncertainty of the data set. The results show that GP-PACC almost exactly mimics the driver in a purely data-driven way, and hence improves the personalization in ADAS by adapting the longitudinal driving assistance to the driver's preferences and needs.

5.4.2. Human-In-The-Loop Experiments on the Unity Game Engine

Modeling and simulation environment in Unity game engine Game engines are conceptually the core software necessary for a game program to properly run. They generally consist of a rendering engine for graphics, a physics engine for collision detection and response, and a scene

Figure 5.4: Naturalistic driving in a car-following scenario with a gaming laptop, a Logitech racing wheel, and the Unity game engine.

graph for the management of elements like models, sound, scripting, threading, etc. Along with the rapid development of game engines in recent years, they become popular options in the development of intelligent vehicle technology [132], with studies conducted for driver behavior modeling [251], connected vehicle systems prototyping [250, 129], and autonomous driving simulation [50, 182].

In this study, human-in-the-loop experiments are conducted on a customized driving simulator platform, which is built with a Windows gaming laptop (processor Intel Core i7-9750 @2.60 GHz, 32.0 GB memory, NVIDIA Quadro RTX 5000 Max-Q graphics card), a Logitech G29 Driving Force racing wheel, and Unity game engine 2019.2.11f1 [252]. A three-lane highway scene is built in the simulation environment, where human drivers are able to manually drive the ego vehicle to follow the target vehicle, shown as Fig. 5.4.

Data acquisition The experiment trip resembles a freeway high-speed scenario, and has a total period of 200 seconds. The lead vehicle's trajectory comes from the CAN-bus data of a pre-recorded trip by a human driver [242]. The trajectory contains a time-varying speed profile within the range 25-35m/s that captures a naturalistic freeway acceleration and deceleration scenario. The data is recorded in 10Hz. The training input and target are organized according to (5.4) and (5.5), where $\mathbf{Z} = \{\mathbf{z}_k = [s, v, u]_k\}_{k=1}^{N}$, and $\mathbf{y} = \{y_k\}_{k=1}^{N}$.

Training result The parameter inference takes about 10 seconds to complete, with the best-estimated parameters $\theta = [l_1, l_2, l_3, \sigma_f, \sigma_n] = [14.4, 1.4, 5.9, 0.56, 0.11]$, where l_1, l_2, l_3 correspond to the characteristic length scales of s, v, u, respectively.

To visualize the training result, Fig. 5.5 compares the GP simulated acceleration and Unity recorded acceleration. The mean prediction (red line) aligns well with the recorded data (dotted black line) both in the training and validation sets. The uncertainty captures the variation of the recorded data in the training set, and accurately acknowledges the uncertain prediction in the validation set

Table 5.1: Model training results: all trained on the same training set and validated on the same validation set shown in Fig. 5.5.

Model	GP	CTH-RV	IDM
MSE - acceleration	0.0909	**0.0742**	0.0927
MSE - velocity	0.851	**0.779**	6.044
MSE - space gap	**24.8**	41.6	1101
LPD - acceleration	-0.0023	N/A	N/A

(with a wider prediction variance), with a few exceptions at around 120 sec.

To further validate that the GP model captures the driving dynamics, we compare its ability to reconstruct human-driving profiles with that of two ODE-based car-following models. The first model is the *constant-time headway relative-velocity* (CTH-RV) model used to characterize adaptive cruise control driving behaviors [81], and the second one is IDM [208], which is used to describe human-driving behaviors. Since the GP-PACC design problem is formulated as a system identification problem, where the goal is to minimize the discrepancy between the predicted driving profiles and the measured ones, it shares the same objective with calibrating car-following models. GP-PACC, CTH-RV and IDM are trained (calibrated) with the same training data, and validated with the same testing data produced from the same driver shown in Fig. 5.5. We use acceleration as the prediction target in order to be consistent with the GP model training, which also has the target of minimizing the error on predicted acceleration. A general form of calibrating any car-following model is written as minimizing a sum-of-squared cost function:

$$\begin{aligned} \underset{\theta}{\text{minimize}}: \quad & \textstyle\sum_{k=1}^{N}(y_k - \hat{y}_k)^2 \\ \text{subject to:} \quad & \hat{y}_k = f_{\text{CF}}(s_k, v_k, u_k, \theta), k = 1, 2, .., N \end{aligned} \tag{5.17}$$

with possible additional constraints on the initial conditions, and bounds on the parameters.

The training and testing errors are measured by MSE on the acceleration and space gap. As shown in Table 5.1, GP can perform on par with, or even outperform some established analytical car-following models in terms of reproducing acceleration, velocity and space gap trajectories. Notably, GP outperforms both other models with the lowest space-gap MSE, which tends to accumulate from inaccurate acceleration prediction.

In addition, we see that the training on naturalistic driving data does not provide satisfactory results as compared to training with synthetic data. One immediate reason is that synthetic data generated using ODE-based models has a cleaner relationship between the inputs (s, v, u) and the output (acceleration), which can be captured by the squared-exponential covariance function (5.3); On the other hand, naturalistic driving data contains more randomness and inconsistent patterns even during the same trip. More driving datasets are desired to test the ability of our model on recovering the longitudinal car-following behavior. It will be interesting to compare not only with other car-following models, but also across datasets (e.g., naturalistic driving datasets [98] and field experimental data for ACC vehicles [82, 135]) for future work. Nevertheless, GP modeling of human-in-the-loop experiments shows promising results, even with no explicit assumptions on the personalized driving styles.

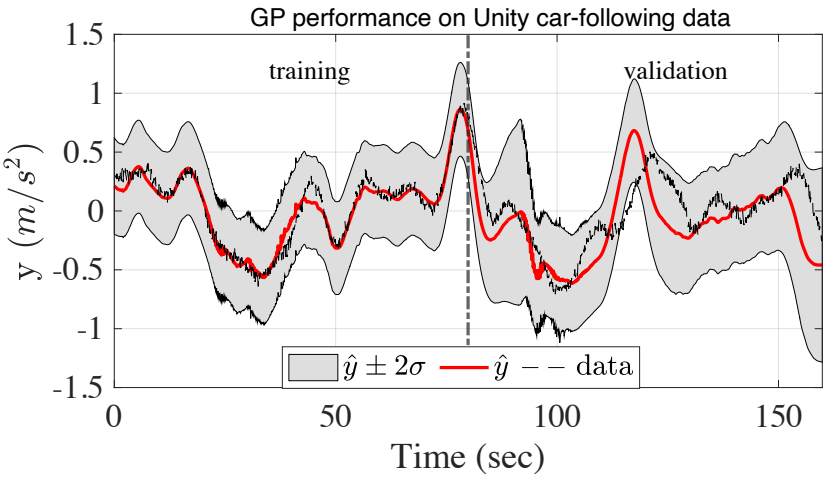


Figure 5.5: Compare GP-PACC guided acceleration (red) with the actual acceleration recorded by one of the human-in-the-loop experiments (dotted black). The first half is training result and the second half is validation result.

5.4.3. **Human-in-the-loop override validation**

In addition to the numerical analysis, the GP controller is also validated with human-in-the-loop override validation. The purpose of the tests is to measure each driver's comfort and trust of the proposed GP-PACC as well as two other baseline models (i.e., CTH-RV and IDM). The test drivers undergo several blind tests: an unknown controller drives the ego vehicle for each trip, and the frequency and duration of which the drivers override the equipped ACC (by stepping on the acceleration/braking pedals) are recorded.

Experiment setup In this validation, instead of using manual control for car-following, the ego vehicle is driven with the trained GP-PACC as well as two other baseline ACC models. Four drivers (two males and two females with diverse real-world driving experience) participate in the tests, and each is randomly provided with the individualized GP-PACC or either of the two baseline models. The operating controller for a specific trip is unknown to the driver in order to eliminate potential bias. Each driver completes the tests when all three controllers are covered.

Each trip lasts 200 sec, where each driver monitors the trip and overrides the equipped ACC when he/she feels uncomfortable. The equipped ACC resumes control immediately after the driver lets go the overrides. The timestamps of which the driver overrides the ACC are be recorded.

Controller specifications GP-PACC is customized for each driver. First, a 200-sec naturalistic car-following data is collected from each driver with the same simulation setup: all drivers are told to naturally follow the same leader, whose speed and acceleration profiles are shown as the blue lines in Fig. 5.6. All trips are recorded on the same Unity game engine with the same Logitech G29 Driving Force racing wheel. Other simulation parameters (e.g., weather, surrounding traffic and road conditions are fixed for all trips). Next, the training for GP-PACC is conducted using Algorithm 2. The resulting GP-PACC specifications are summarized in Table 5.2.

The other two baseline ACC models are taken directly from two calibrated ACC models. Specifically,

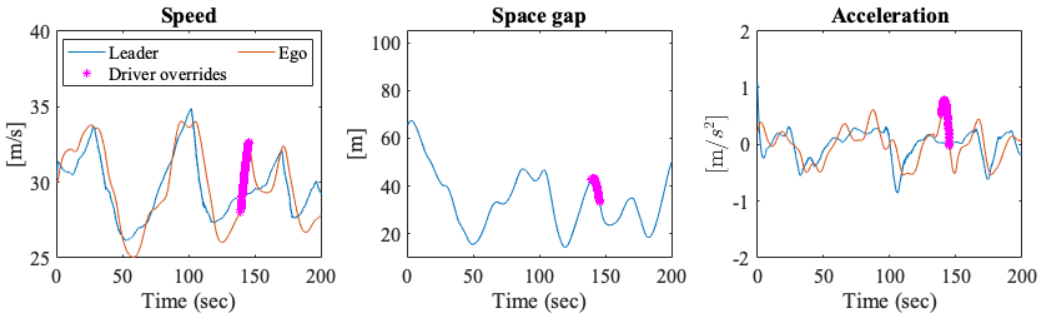


Figure 5.6: A trip driven by GP controller with driver B behind the wheel.

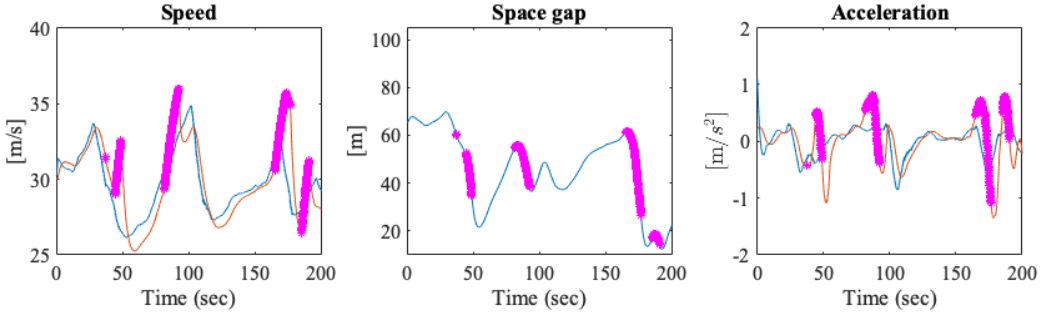


Figure 5.7: A trip driven by ACC#1 with driver B behind the wheel.

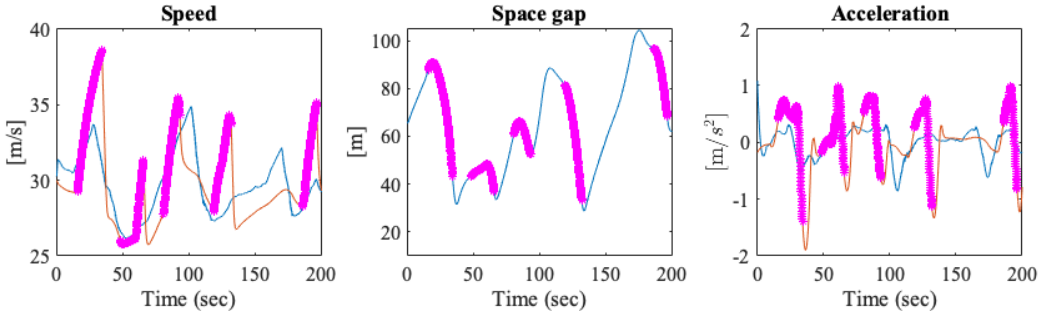


Figure 5.8: A trip driven by ACC#2 with driver B behind the wheel.

Table 5.2: GP-PACC parameters for each driver

Driver	l_1	l_2	l_3	σ_f	σ_n
A	14.4	1.40	5.90	0.56	0.11
B	18.4	1.72	1.11	0.43	0.20
C	5.61	0.47	1.70	0.56	0.20
D	4.33	1.25	2.40	0.17	0.19

ACC#1 is the constant-time headway relative-velocity (CTH-RV) model of the form:

$$a_k = 0.0131(s_k - 1.6881v_k - 7.57) + 0.2692(u_k - v_k), \tag{5.18}$$

and ACC#2 is of the form of an IDM:

$$a_k = 0.73\left[1 - \left(\frac{v_k}{30}\right)^4 - \left(\frac{s^*(v_k, u_k)}{s_k}\right)^2\right], \tag{5.19}$$

where the desired space gap s^* is defined as:

$$s^*(v_k, u_k) = 2 + 1.5v_k + \frac{v_k(v_k - u_k)}{2.21}. \tag{5.20}$$

The parameters for both baseline ACCs are chosen as suggested in [81, 208].

Results All four drivers override the operating ACC models to different extents. From Table 5.3, in general, all drivers intervene the vehicle less when running GP-PACC as compared to running other two baseline ACC models. On average, all drivers override only 4.43% (8.7 sec total) of the 200-sec trip when GP-PACC is on board.

As an illustration, the recorded trips from one of the drivers (driver B) can be visualized in Fig. 5.6-5.8. The top row (Fig. 5.6) shows the trajectories when GP-PACC is the selected controller. The middle row (Fig. 5.7) corresponds to ACC#1 (CTH-RV controller) being in operation and the bottom row (Fig. 5.8) corresponds to ACC#2 (IDM controller). The recorded trajectories include the speeds for leader and follower (left-most column), space gap (middle column) and accelerations for leader adn follower (right column) with respect to time. The magenta highlights indicate the timestamps when the driver overrides ACC (either by pressing gas or brake pedal) that is in operation.

Driver B indicates that he overrides when he feels “falling behind from the lead vehicle, and the neighboring vehicles on the right lane will cut into the gap between the ego vehicle and the lead vehicle". Fig. 5.6 shows that the driver feels comfortable when GP-PACC is in control, i.e., the driver only overtook the controller for about 1 second during the entire trip. On the other hand, the driver pressed the gas pedal several times when ACC#1 is in operation (Fig. 5.7), and even more so with ACC#2 engaged (Fig. 5.8). The results strongly indicate that the driver favors the personalized controller (GP-PACC) in the unbiased test settings.

Table 5.3: Human-in-the-loop experiments results: drivers gas & brake takeover percentage during a 200-sec trip

Driver	GP-PACC	ACC#1(CTH-RV)	ACC#2(IDM)
	Gas / Brake	Gas / Brake	Gas / Brake
A (F)	**0% / 0.60%**	4.2% / 0.65%	11.9% / 0.6%
B (M)	**3.1% / 0.15%**	14.6% / 1.8%	26.7% / 1.7%
C (F)	**8.6%/ 2.7%**	18.8% / 4.6%	58.8% / 8.25%
D (M)	**2.3% / 0.25%**	4.4% / 0.35%	22.4% / 0.9%
Avg.	**3.5% / 0.93%**	10.5% / 1.85%	30.0% / 2.86%

5.5. **Conclusion and Future Work**

In this chapter we propose GP-PACC that mimics personalized car-following behavior. The learning is achieved using a Gaussian Process regression with nonlinear output-error training on the car-following data. We explore this purely data-driven controller design to capture personalized driving styles, which sometimes cannot be captured by an explicit car-following model.

The training result shows that GP has the potential to provide safe and realistic acceleration guidance that closely resembles personalized acceleration profile. Specifically, GP almost exactly recovers the car-following profiles of an IDM driver (data generated using an IDM), and outperforms two other established analytical car-following models in terms of reproducing naturalistic car-following space gap trajectories. A series of human-in-the-loop experiments are conducted on the Unity driving simulator to test drivers' override rates when running their personalized GP-PACC versus other baseline ACC models. Results indicate that all tested drivers express comfort using GP-PACC, which reduces the human override duration 60% and 85% as compared to two other standard ACC models, respectively. This brings promising potentials of the acceptance towards the personalized controller in near real-world scenarios.

For future work, adaptive GP training can be incorporated into current routine to enhance the proposed GP-PACC. Since training a GP dynamical system requires only limited data, it is possible to adaptively train the GP model as more data is collected. This training procedure allows to capture the variations in driving behaviors across a longer period of time and a wider range of speed. Additionally, since driver override has only been adopted as a measurement to test our GP-PACC in this chapter, it can also be considered as a direct feedback to the GP model, which will enhance the performance of our future GP-PACC in a more straightforward manner. As more contextual information (e.g., weather, road geometry) becomes available, the training features can be augmented to expand the functionality of personalization.

CHAPTER 6

Macroscopic traffic dynamics and state estimation

6.1. Introduction

In this section we focus on the macroscopic scale traffic analysis – instead of acquiring measurements and modeling the driving behavior of individual vehicles, we concern about the evolution of overall traffic flow and density. The macro-scale analysis allows us to see the evolution of traffic patterns and provide insights towards infrastructure-level monitoring and control in order to, for example, mitigate congestion.

In addition to the complexity introduced by the interaction between human-driven vehicles and automated vehicles, another dimension of complexity is due to the diverse vehicle types and the distinct driving rules associated with each class, which is still prevalent in developing countries. There are several challenges when studying the mixed traffic environment: (1) existing macroscopic traffic models are insufficient to describe the complexity in non-trivial vehicular interactions; (2) macroscopic traffic data are often noisy and extremely sparse; (3) although there has been an increasing amount of research efforts to study macroscopic traffic evolution by combining models and measurements through estimation approaches, the complexity in mixed traffic often hinders the estimation accuracy.

In this chapter and the next we consider the traffic state estimation problem with a case study in heterogeneous (multi-class) traffic. To address the challenges above, we provide the following contributions:

- **Multi-class traffic estimation formulation** We leverage a previously developed multi-class traffic flow model that can capture complex vehicular interactions, to formulate the state estimation problem where the density evolution of each vehicle class is estimated simultaneously.
- **New particle filter algorithms** We propose several novel algorithmic designs based on the particle filters to estimate the complex evolution of mixed traffic given sparse and noisy data. The proposed algorithms are applied to nonlinear and non-differentiable state evolution dynamics.
- **Validation with real traffic data** The proposed traffic estimation techniques are demonstrated using measurement data collected at a mixed-traffic corridor. We show that the proposed methods can accurately recover the traffic density evolution for each vehicle class.

This chapter begins with a brief introduction to traffic flow models in Section 2.2. The formulation of traffic state estimation problem is described in Section 6.3 with a focus on particles filters

6.2. Overview on a two-class creeping model

The two-class creeping model [56] is a system of scalar conservation laws that governs the flow of each vehicle class:

$$\frac{\partial \rho_j(x,t)}{\partial t} + \frac{\partial \rho_j(x,t) V_j(r(x,t))}{\partial x} = 0,\ j \in \{1,2\}, \tag{6.1}$$

where $\rho_j(x,t)$ denotes the density of each vehicle class (indexed by j) at time t and space x. The velocity function for each class $V_j(\cdot)$ is distinct for each vehicle class and depends on the total density

$r = \sum_j \rho_j$. For simplicity, in this work we consider the following velocity functions:

$$V_j(r) = \max\left(v^m\left(1 - \frac{r}{r_j^m}\right), 0\right),\ j \in \{1, 2\} \tag{6.2}$$

where v^m is the speed limit applied to all road users. The class specific jam densities $r_j^m \in \{r_1^m, r_2^m\}$ control the total density r at which the individual vehicle classes come to a complete stop. If $r_1^m \neq r_2^m$, then one vehicle class will be able to creep through traffic while the other class is completely stopped. In the simplified setting considering a piecewise linear velocity function, the three parameters, v^m, r_1^m and r_2^m, completely define the two class creeping flow. Note that the creeping model is well posed [56], which is, in general, difficult to establish for many macroscopic models in which overtaking occurs.

Note that the model is able to capture a variety of traffic regimes such as overtaking (i.e., faster vehicles overtaking slower ones) and creeping (small vehicles overtaking large vehicles that have come to a complete stop). In multi-class traffic, the traffic regimes can be further complicated, for example, when one class is congested but the other class remains in free flow. More traffic regimes are tested in the numerical experiments in [241].

A numerical scheme is used to approximate the solution to the PDE (6.1) based on the Godunov scheme [76], which requires solving a Riemann problem at every interface between each pair of consecutive and discretised road segments at each time step. On scalar models, the approach leads to the well known *cell transmission model* (CTM) [46, 47]. The discretised creeping model reads as follows:

$$\rho_{i,j}^{k+1} = \rho_{i,j}^k + \frac{\Delta t}{\Delta x}\left(F_{i-\frac{1}{2},j}^k - F_{i+\frac{1}{2},j}^k\right),\ j \in \{1, 2\}, \tag{6.3}$$

where $\rho_{i,j}^k$ represents the density of class j in the ith cell at time k. The terms $F_{i-1/2,j}^k$ and $F_{i+1/2,j}^k$ are the numerical fluxes of class j via the upstream and downstream boundaries of cell i at time k.

For simplicity of the notation, we use subscripts on variables, e.g., $\rho_{-,j}$ and $\rho_{+,j}$, to represent upstream and downstream densities respectively of class j. The flux for vehicle class j over a cell boundary is thus defined as:

$$\begin{aligned} &F_j(\rho_{-,1}, \rho_{-,2}, \rho_{+,1}, \rho_{+,2}) \\ &\quad = \min\{S_j(\rho_{-,1}, \rho_{-,2}), R_j(\rho_{+,1}, \rho_{+,2})\},\ j \in \{1, 2\}, \end{aligned} \tag{6.4}$$

where $S_j(\cdot, \cdot)$ and $R_j(\cdot, \cdot)$ are the sending and receiving functions for vehicle class j defined as:

$$S_j(\rho_{-,1}, \rho_{-,2}) = \begin{cases} Q_j(\rho_{-,1}, \rho_{-,2}) & \text{if } \rho_{-,j} \leq \rho_j^c(\rho_{-,\hat{j}}) \\ Q_j^{\max}(\rho_{-,\hat{j}}) & \text{if } \rho_{-,j} > \rho_j^c(\rho_{-,\hat{j}}) \end{cases} \tag{6.5}$$

$$R_j(\rho_{+,1}, \rho_{+,2}) = \begin{cases} Q_j^{\max}(\rho_{+,\hat{j}}) & \text{if } \rho_{+,j} > \rho_j^c(\rho_{+,\hat{j}}) \\ Q_j(\rho_{+,1}, \rho_{+,2}) & \text{if } \rho_{+,j} \leq \rho_j^c(\rho_{+,\hat{j}}), \end{cases} \tag{6.6}$$

where $\rho_{\hat{j}}$ denotes the density of the other vehicle class. In addition, $Q_j(\rho_1, \rho_2) = \max\{\rho_j V_j(\rho_1 + \rho_2), 0\}$, $Q_1^{\max}(\rho_2) = \max_{\rho_1} Q_1(\rho_1, \rho_2)$ and $\rho_1^c(\rho) = \frac{r_1^m - \rho_2}{2}$ is the critical density of ρ_1 such that $Q_1^{\max}$ is obtained. Similarly, $Q_2^{\max}(\rho_1) = \max_{\rho_2} Q_2(\rho_1, \rho_2)$ and $\rho_2^c(\rho) = \frac{r_2^m - \rho_1}{2}$ is the critical density of ρ_2 such that $Q_2^{\max}$ is obtained.

For a complete description and analysis of the model, the reader is referred to [56].

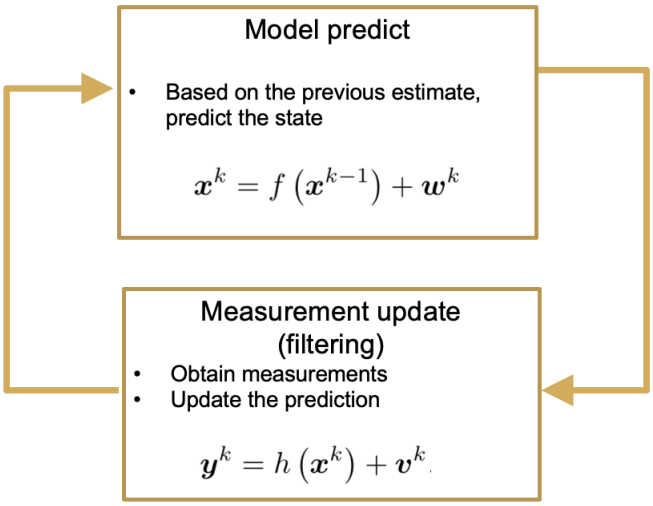


Figure 6.1: State estimation procedure

6.3. **Bayesian traffic state estimation**

Building on these works, we consider the problem of traffic state estimation when the base traffic flow is heterogeneous and nontrivial interactions such as overtaking occur between classes. The traffic flow dynamics are described by the two-class creeping model [56], which allows small vehicles (e.g., motorbikes) to overtake larger ones, including when the large vehicles come to a complete stop. Traffic state estimation is performed using one of several fully nonlinear particle filtering algorithms. Because heterogeneous traffic may have significantly different operating rules compared to a homogeneous passenger-car traffic environment, we investigate the ability of each of the particle filters to reconstruct multi-class traffic in both simulated and real settings where overtaking of large vehicles by small vehicles can occur, particularly in highly congested traffic.

The Bayesian approach to traffic state estimation evaluates the posterior distribution of the system state given a prior state estimate and measurement data. A diagram is shown in Figure 6.1. At each timestep, a prediction distribution is obtained based on a Markovian-like process $f(\cdot)$, and the distribution is re-weighted based on the newly-obtained measurement distribution y. This model-prediction and measurement-correction loop continues sequentially.

The state of the system $\boldsymbol{x}^k$ for model (6.3) at time k is defined as:

$$\boldsymbol{x}^k = \left[\rho^k_{1,1}, \ldots, \rho^k_{i_{\max},1}, \rho^k_{1,2}, \ldots, \rho^k_{i_{\max},2}\right]^T, \tag{6.7}$$

where $i_{\max}$ is the number of cells in the spatial discretisation.

The state propagation equation is:

$$\boldsymbol{x}^k = f(\boldsymbol{x}^{k-1}, \boldsymbol{\theta}) + \boldsymbol{w}^k, \tag{6.8}$$

where $f(\cdot,\cdot)$ is the discrete-time creeping model defined in (6.3), and it propagates the traffic state to the next time step, with the input parameter vector $\boldsymbol{\theta} = [v^m, r_1^m, r_2^m]^T$. The measurement equation is:

$$\boldsymbol{y}^k = h(\boldsymbol{x}^k) + \boldsymbol{v}^k, \tag{6.9}$$

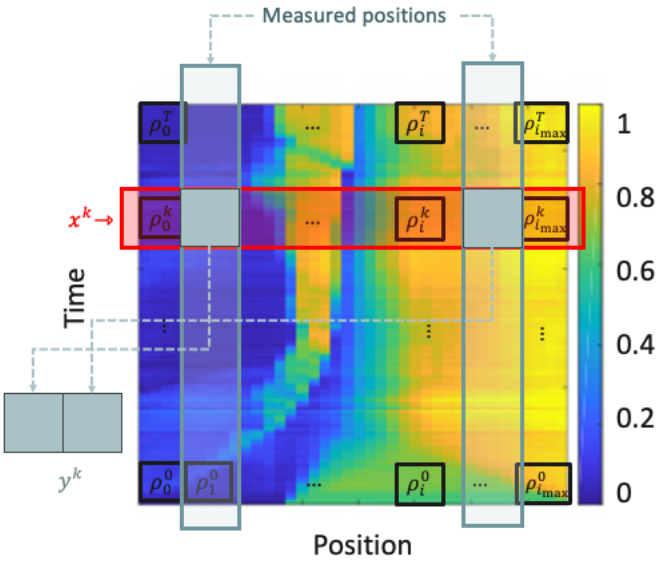


Figure 6.2: Illustration of traffic state estimation with the evolution of traffic density over time and space. Traffic density ρ_i^k represents location i at time k. Sparse measurements are taken at specific locations on the roadway. The state vector x^k represents densities at all positions at time k, while partial observation is denoted as y^k.

where $\boldsymbol{y}^k$ is the sensor data obtained at time k and relates to the system state through the measurement equation $h(\cdot)$. In the case when (a subset of) the system state is directly measured, the observation equation is linear. The terms $\boldsymbol{w}^k \sim \mathcal{N}(0, Q)$ and $\boldsymbol{v}^k \sim \mathcal{N}(0, R)$ denote the additive unbiased process noise and measurement noise at time k with assumed covariance matrices Q and R.

The state estimation problem can be viewed as sequentially evaluating the prior state distribution $p(\boldsymbol{x}^k|\boldsymbol{Y}^{k-1})$ and the posterior state distribution $p(\boldsymbol{x}^k|\boldsymbol{Y}^k)$ given measurements $\boldsymbol{Y}^k = [\boldsymbol{y}^1, \boldsymbol{y}^2, \ldots, \boldsymbol{y}^k]$, according to:

$$\begin{aligned} p(\boldsymbol{x}^k|\boldsymbol{Y}^{k-1}) &= \int p(\boldsymbol{x}^k|\boldsymbol{x}^{k-1}) p(\boldsymbol{x}^{k-1}|\boldsymbol{Y}^{k-1}) d\boldsymbol{x}^{k-1} \\ p(\boldsymbol{x}^k|\boldsymbol{Y}^k) &= \frac{p(\boldsymbol{y}^k|\boldsymbol{x}^k) p(\boldsymbol{x}^k|\boldsymbol{Y}^{k-1})}{p(\boldsymbol{y}^k|\boldsymbol{Y}^{k-1})}. \end{aligned} \tag{6.10}$$

In the particle filter described next in Section 6.4, the probability distributions (6.10) are evaluated based on sequential Monte Carlo sampling.

6.4. **Filters design**

In this section, we summarise the standard bootstrap *particle filter* (PF) and discuss the weight degeneracy as measured by the *effective particle size*. Then we propose two enhancement methods, namely parameter-adaptive filtering and spatially-correlated process noise modelling, to improve the PF estimation.

Due to the nonlinearity and non-differentiability of the process model (6.1), discontinuities in the traffic state can occur, which can generate a multi-modal state distribution and limit the performance of minimal variance estimators such as EKF and UKF. The challenge motivates the use of PF, with the idea of propagating and updating Monte Carlo samples sequentially to represent the full state distribution without restrictive assumptions on the system dynamics and the noise distribution.

The particle filter starts with a collection of N_p samples (referred to as particles) from the initial state probability density function $p(\boldsymbol{x}^0)$, where $\boldsymbol{x}^0$ is a random variable representing the state vector at time $k = 0$. Each realisation of the state vector is denoted as $\boldsymbol{x}_l^0$, $l = 1, \cdots, N_p$. At each time instant, the particles are propagated to the next time step using the discrete time process model $f(\boldsymbol{x}, \boldsymbol{\theta})$, i.e., the traffic flow model (6.3), to approximate the prior state distribution $\boldsymbol{x}^{k|k-1}$ at time k.

The state distribution is updated after measurements are obtained. Specifically, a weight (q_l) is assigned to each particle based on the conditional relative likelihood evaluated from the likelihood function $p(\boldsymbol{y}^k|\boldsymbol{x}^k)$. The posterior state distribution $\boldsymbol{x}^{k|k}$ is approximated by resampling the particles according to the new weight distribution. This step ensures that the heavier-weighted particles are more likely to be drawn from the probability density function while the total number of particles is preserved.

Algorithm 3 PF algorithm for traffic state estimation

Initialise: Draw $\boldsymbol{x}_l^{0|0}$ from $\mathcal{N}(\boldsymbol{\mu}^0, Q^0)$ for $l = 1 : N_p$
for k = *1:T* **do**
  State propagation:
  $\boldsymbol{x}_l^{k|k-1} = f(\boldsymbol{x}_l^{k-1|k-1}, \boldsymbol{\theta}) + \boldsymbol{w}_l^k$ for all l
  State update:
  Assign weight: $q_l \coloneqq p[(\boldsymbol{y}^k = \boldsymbol{y}^{k*})|(\boldsymbol{x}^k = \boldsymbol{x}_l^{k|k-1})]$
  Normalise weight: $q_l \coloneqq \frac{q_l}{\sum_{l=1}^{N_p} q_l}$
  Resample:
  Draw $\boldsymbol{x}_l^{k|k}$ with probability q_l
end
$\boldsymbol{\mu}^0$: mean of the initial state distribution
Q^0: initial state covariance matrix
$\boldsymbol{x}_l^{k|k-1}$: sample l from prior state distribution at time k
$\boldsymbol{x}_l^{k|k}$: sample l from posterior state distribution at time k
$\boldsymbol{w}_l^k$: a realisation of the process noise $\boldsymbol{w}^k \sim \mathcal{N}(0, Q)$, where Q is the covariance matrix of $\boldsymbol{w}^k$
$\boldsymbol{y}^{k*}$: a measurement at time k

6.4.1. Parameter-adaptive particle filtering (PAPF)

Inspired by the dual filtering approach for simultaneous state and model parameter estimation [86, 213, 162], in this filter, we allow the estimated model parameters to be adjusted at each time step instead of having fixed values, i.e., we model the parameters as time-invariant (the dynamics do not change over time) with some noise as approached in the standard dual-filtering problems mentioned above. This gives the estimator extra flexibility that can potentially produce more accurate state

estimates. The goal is not for online parameter estimation due to the challenge in identifiability analysis of a nonlinear and non-differentiable model, but simply allowing parameter estimates to be updated in motion to facilitate state estimation. We name this approach *parameter-adaptive particle filter*, or PAPF in the remaining of this chapter.

The PAPF includes an additional particle filter running in parallel with the state estimator to adjust the estimated parameters. In the parameter propagation step, the parameter samples are obtained by performing a random walk from the best estimated parameter in the previous timestep, $\hat{\boldsymbol{\theta}}^{k-1|k-1}$. In the parameter update step, the prior state distribution ($\boldsymbol{x}^{k|k-1}$) is approximated by propagating the best state estimate at the previous timestep ($\hat{\boldsymbol{x}}^{k-1|k-1}$) through the traffic flow model (6.3) with the parameter samples. The remaining parameter update step follows a similar approach described in Algorithm 3: each parameter sample is re-weighted according to the relative likelihood function $p(\boldsymbol{y}^k|\boldsymbol{x}^k)$ after measurements are obtained, and resampled according to the new weight distribution. The estimator of the posterior parameter distribution, $\hat{\boldsymbol{\theta}}^{k|k}$, proceeds next for the state update. The state estimation exactly follows Algorithm 3 with the exception that the parameter in the state propagation equation is now the best posterior parameter estimator, $\hat{\boldsymbol{\theta}}^{k|k}$, instead of a deterministic parameter $\boldsymbol{\theta}$. The detailed PAPF algorithm is summarised in Algorithm 4.

6.4.2. **Spatially correlated noise modelling (SCNM)**

This approach differs from Algorithm 3 in terms of the process noise at time k, $\boldsymbol{w}^k \sim \mathcal{N}(0, Q)$. In the PF, we apply the commonly implemented assumption that Q is a diagonal matrix, indicating that the elements of the state vector $\boldsymbol{x}^k$ are uncorrelated. It is suggested by [25] that if the traffic in one cell is extremely congested, then the vehicles interact very often with each other, and their location and speed will be highly correlated. Motivated by this observation, we modify the covariance matrix of the process noise $\boldsymbol{w}^k$ to $Q(i, i')$ with off-diagonal terms, which represents the similarity between all possible pairs of cells (indexed by i and i'). We use a covariance expression $Q(i, i') = \exp\left(-\frac{|i-i'|}{d}\right) \times \sigma^2_{\boldsymbol{w}^k}$, where $\sigma_{\boldsymbol{w}^k}$ is the standard deviation of noise $\boldsymbol{w}^k$. The characteristic length-scale d is a measure of how far away two cells (i, i') need to be for the cell values to be uncorrelated. The correlation between two cells is assumed to depend solely on the relative distance of the pair instead of the absolute location of the cells.

In practice, the traffic state can be highly correlated in space, i.e., cells in freeflow traffic are likely to occur next to each other, and similarly with the congested flow. We encode this heuristic via correlation in the process noise to account for a similar traffic pattern in neighbouring cells, and decrease the correlation with respect to the relative distance as one would expect. The introduction of a spatially correlated process noise increases the correlation on the prior state distribution, and is shown in [241] to improve the effective particle size.

6.5. **A case study: heterogeneous traffic state estimation**

In this section, we apply the enhanced particle filters on heterogeneous trajectory data collected in Chennai, India [101]. We first briefly describe the dataset and preparation for the use of our proposed methods. Then, we describe the experiment setup including the estimated model parameters and particle filter parameters. Finally, we discuss the results.

Algorithm 4 PAPF algorithm

Initialise:

Draw $\boldsymbol{x}_l^{0|0}$ from $\mathcal{N}(\boldsymbol{\mu}^0, Q^0)$ for $l = 1 : N_p$

Set $\hat{\boldsymbol{x}}^{0|0} = \boldsymbol{\mu}^0$ and $\hat{\boldsymbol{\theta}}^{0|0} = \boldsymbol{\theta}^0$

for k = *1:T* **do**

Parameter propagation:

$\boldsymbol{\theta}_m^{k|k-1} = \hat{\boldsymbol{\theta}}^{k-1|k-1} + \boldsymbol{\eta}_m^k$ for $m = 1 : N_m$

Parameter update:

$\boldsymbol{x}_m^{k|k-1} = f(\hat{\boldsymbol{x}}^{k-1|k-1}, \boldsymbol{\theta}_m^{k|k-1}) + \boldsymbol{w}_m^k$ for all m

Assign weight: $q_m \coloneqq p[(\boldsymbol{y}^k = \boldsymbol{y}^{k*})|(\boldsymbol{x}^k = \boldsymbol{x}_m^{k|k-1})]$

Normalise weight: $q_m \coloneqq \frac{q_m}{\sum_{l=1}^{N_m} q_m}$

Resample:

Draw $\boldsymbol{\theta}_m^{k|k}$ with probability q_m

Update $\hat{\boldsymbol{\theta}}^{k|k}$

State propagation:

$\boldsymbol{x}_l^{k|k-1} = f(\boldsymbol{x}_l^{k-1|k-1}, \hat{\boldsymbol{\theta}}^{k|k}) + \boldsymbol{w}_l^k$ for all l

State update:

Assign weight: $q_l \coloneqq p[(\boldsymbol{y}^k = \boldsymbol{y}^{k*})|(\boldsymbol{x}^k = \boldsymbol{x}_l^{k|k-1})]$

Normalise weight: $q_l \coloneqq \frac{q_l}{\sum_{l=1}^{N_p} q_l}$

Resample:

Draw $\boldsymbol{x}_l^{k|k}$ with probability q_l

Update $\hat{\boldsymbol{x}}^{k|k}$

end

$\boldsymbol{\mu}^0$: mean of the initial state distribution

Q^0: initial state covariance matrix

$\boldsymbol{\theta}^0$: initial parameter values

$\boldsymbol{\eta}_m^k$: a realisation of the parameter noise $\boldsymbol{\eta}^k \sim \mathcal{N}(0, Q_\theta)$, where Q_θ is the covariance matrix of $\boldsymbol{\eta}^k$

N_m: number of parameter samples at each time step

$\hat{\boldsymbol{x}}^{k|k}$: a point estimate of the state at time k

$\hat{\boldsymbol{\theta}}^{k|k}$: a point estimate of the parameter at time k

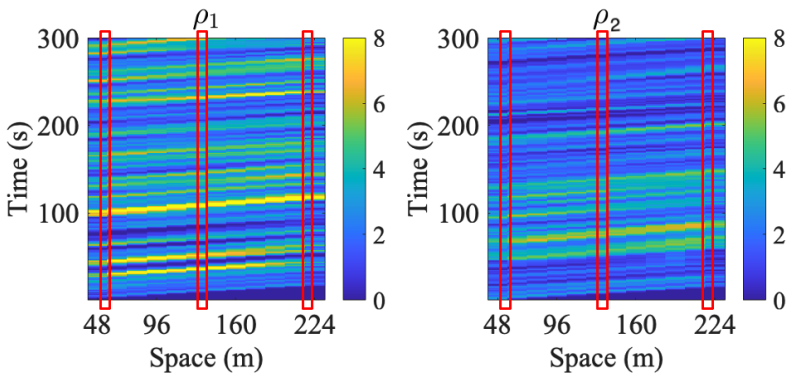


Figure 6.3: Density evolution of real heterogeneous traffic data. The red rectangles indicate measurement positions.

6.5.1. **Data description and preparation**

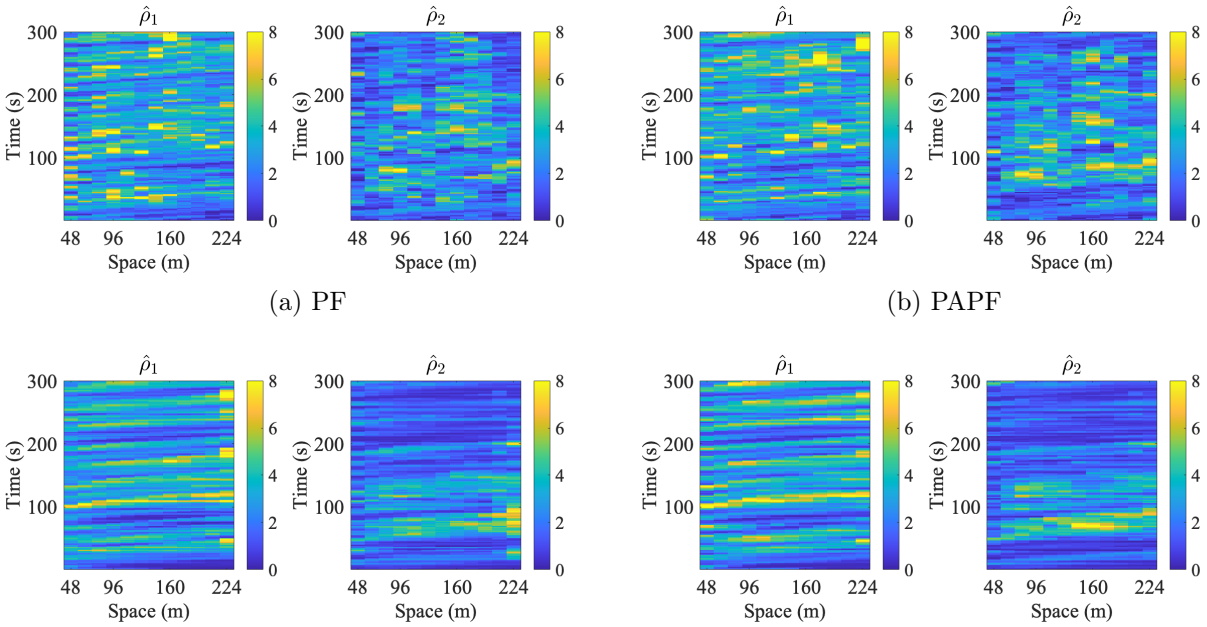
In this study, we incorporate the vehicle trajectory data in mixed traffic [101]. The dataset was extracted from the video sequences in an urban midblock road section in Chennai, India. Various types of road users were present in the data, such as passenger vehicles, buses, motorbikes and auto-rickshaws. The data was prepossessed to include 3,005 vehicle trajectories, and the positions were recorded at a resolution of 0.5 s for 15 min on a stretch of 245 m, 3-lane city roadway. The total traffic flow observed in the study section is 6,010 vehicles per hour, and the instantaneous speeds vary from 0 to 15.22 m/s.

For the purpose of this study, we first discretize the densities such that $\Delta x = 16.3m$ (or $i_{max} = 15$) and $\Delta t = 1s$. We then count the number of occurrences of each vehicle in each discretised cell at each time step. We separate the counts of motorbikes (as the smaller, faster class, corresponding to ρ_1) and the counts of all other road agents combined (or the larger, slower class corresponding to ρ_2), because of motorbikes' observed overtaking properties. Lastly, a kernel density estimation (KDE) approach [26, 89] with a fixed Gaussian kernel is employed to transform the initial counts into macroscopic traffic quantities (e.g., density of each vehicle type, in number of vehicles per cell) across time and space. We use $\rho^k_{i,j}$ to denote the density of class j at cell i at time step k.

The obtained heterogeneous traffic density data can be visualised in Figure 6.3. It also serves as the 'ground truth' macroscopic traffic data for the state reconstruction tasks.

6.5.2. **Experiment setup**

The roadway is discretised into $i_{\max} = 15$ cells and the experiments are run for $k_{\max} = 300$ time steps, or 300 seconds. All four PFs use the same approxiamte model, with the specifications summarised in Table 6.2. Again for filters with parameter-adjustment (PAPF and PAPF+SCNM), the approximate model parameters are the initial parameter estimates, and the noise on each parameter is assumed to follow a Gaussian zero mean and 0.005 standard deviation. In addition, both the parameter-update step and the state-update step use $N_p = 500$ particles, which is empirically shown as suitable for the state dimension. We assume that the noisy density measurements for both vehicle classes are obtained in an upstream, an intermediate and a downstream cells indexed



(a) PF

(b) PAPF

(c) PF+SCNM

(d) PAPF+SCNM

Figure 6.4: Filter performance: estimated density evolution

by $i = 2, 8$ and 14, as indicated by the red rectangles in Figure 6.3. The initial state noise, the measurement noise and the state prediction noise are all assumed to be Gaussian zero mean, with standard deviation of 1 vehicle/cell. For filters with spatially correlated noise (PF+SCNM and PAPF+SCNM), a characteristic length of 15 is used.

Parameter	Values
Initial conditions	$\hat{\rho}^0_{i,1} = \begin{cases} 4 & i \in [1,8] \\ 1 & \text{otherwise} \end{cases}$
	$\hat{\rho}^0_{i,2} = 1, \forall i$
Boundary conditions	$\hat{\rho}^k_{0,1} = \text{sgn}\left(\sin(0.4\ k)\right) \times 8 + 8$
	$\hat{\rho}^k_{0,2} = 1, \forall k$
	$\hat{\rho}^k_{i_{max}+1,1} = 2, \forall k$
	$\hat{\rho}^k_{i_{max}+1,1} = 2, \forall k$

Table 6.1: Initial and boundary conditions

In addition, we empirically choose the model initial and boundary conditions (Table 6.1) to best represent the observed measurements. The approximate model with the specified parameters, initial and boundary conditions is shown to yield MAEs of 1.92 and 1.67 vehicles/cell for class ρ_1 and ρ_2, respectively, in the studied space and time frame.

Parameter	Values
v^m	15.3 m/s
r_1^m	16 vehicles/cell
r_2^m	10 vehicles/cell

Table 6.2: Filter performance summary on real data.

	Average improvement (MAE reduction %)			
Class	PF	PAPF	PF+SCNM	PAPF+SCNM
ρ_1	31.9	33.2	46.4	43.9
ρ_2	25.8	29.5	46.3	31.5

Table 6.3: Approximate model parameters

6.5.3. **Results and discussion**

The estimated density evolution from PF, PAPF, PF+SCNM and PAPF+SCNM can be visualised in Figure 6.4. A visual inspection indicates that filters with spatial correlation (PF+SCNM and PAPF+SCNM) generally have a more pronounced state reconstruction performance than PF and PAPF. The flow for both vehicle classes is 'smoother' across the space. Practically, spatial correlation in the states help to correct one part of the states, which carries over to its neighbourhoods. It models the similarity in densities of cells of close vicinity, which may implicitly capture the flow dynamics of traffic that PF without SCNM cannot capture.

It can also be observed that PAPF does not provide as significant improvement in traffic state reconstruction as SCNM does. This could be due to a combination of the identifiability issue of the creeping model parameters as well as observability issue of the state, which are not in the scope of this work. Since the recorded traffic is mostly in free-flow state, the filter performance cannot generalize to a variety of traffic regimes. When available, heterogeneous traffic data that contains traffic jams or creeping scenarios should be used to validate our proposed filters for various traffic state reconstruction.

Nevertheless, with the limited heterogeneous data available, the enhanced filters show significant improvement than the standard PF, and improve the estimation accuracy up to 46% as compared to using the approximate model alone. The findings using real heterogeneous data is also in agreement with the results from the numerical experiments: PF with parameter-adaptation and spatially-correlated noises are promising enhancement for traffic state estimation problems.

6.6. **Conclusion**

Considering previous traffic estimation works mainly focus on homogeneous flow with strict lane adherence, this work tackles estimation problem on heterogeneous traffic where non-trivial vehicular interactions occur. Due to the filtering challenges caused by the nonlinear and non-differentiable nature of the traffic flow model, in this chapter we propose three methods to enhance the standard particle filter to estimate complex traffic, both in simulated environments and with real heteroge-

neous traffic data. The results show that the enhanced PFs, especially with spatially-correlated noise modelling, can reduce the estimation error up to 80% and 46% from forward simulation using the approximate model, using the synthetic data and real data, respectively. The enhanced PFs significantly and consistently outperform the standard PF in all scenarios considered.

This chapter is a starting point for further work in the field of heterogeneous traffic state estimation. For example, this work demonstrates that enhanced particle filtering techniques can improve the accuracy of heterogeneous traffic state estimation, and explored the performance as a function of the traffic regime. A rigorous analysis on model observability and/or error boundedness (e.g., extensions to [24] and [198] for heterogeneous traffic models) is challenging but could provide theoretical insights on expected filter performance. Moving towards realistic deployment settings, the functional form of the velocity function will also be important questions to consider. Finally, field data that records more complex heterogeneous traffic scenarios would be insightful to evaluate the performance of the proposed filters when it becomes available.

CHAPTER 7

Bridging micro-macro traffic studies through trajectory data reconstruction

7.1. Introduction

Vehicle trajectory data is integral to the study of traffic dynamics. They reveal the relationship between individual traffic participants and the resulting traffic flow phenomena. The increasing volume of data sources is enabling revolutionary research on, for example, traffic flow theory and corridor management.

Accurate estimation of energy consumption also relies on vehicle-level detailed dynamics [59, 60, 161]. High-quality trajectory data can close the gap for understanding microscopic traffic phenomena, such as lane-change and car-following [68, 71, 184, 106, 207] and the impact of mixed autonomy in traffic [193, 79, 261].

We draw attention to the following necessary features for trajectory data in order to enable modern transportation studies:

- **Large spatial-temporal scale.** Current trajectory datasets have limitations in both spatial and temporal coverage, hindering the calibration and training of traffic flow models. Expanding the scale enables sampling of diverse traffic scenarios, contributing to advancements in traffic flow theory and artificial intelligence algorithms [122].
- **Sufficient microscopic fidelity.** In addition to a large scale, data must possess fine-grained vehicle positions to validate and construct realistic microscopic models. Traditional collection methods like loop detectors provide low-frequency, aggregated features suitable for macroscopic characteristics. However, validating microscopic features without individual vehicle-level data is challenging. Video-based trajectory data, combined with image processing algorithms, can offer a comprehensive "bird's eye" view and achieve complete spatial and temporal coverage of roadway segments.
- **Continuous data generation.** Generating live traffic measurements is crucial for effective traffic control strategies such as variable speed limits, ramp metering, and Lagrangian control with connected and autonomous vehicles. Real-time deployment requires an efficient data reconciliation pipeline capable of handling streaming data with low latency.

Ultimately, vehicle and transportation technologies are trending towards "intelligent" and more autonomous solutions. Being able to measure the broader impact on traffic will be crucial in enabling future generations of traffic control and management. Now with the advancement in camera resolutions and computer-vision algorithms for object detection and tracking, the process of getting reliable trajectory data can be greatly automated, yet issues on the quality of the data still remain. This chapter aims to provide a comprehensive pipeline to systematically rectify trajectory data produced by automatic computer vision algorithms. Leveraging physical constraints such as vehicle dynamics and state information, this pipeline outputs trajectories that significantly improve the validity and quality of the raw data. The core methods presented here are in the process of being deployed on a large-scale open-road testbed – the *I-24 MOTION system*, which we describe later in this chapter.

To address the challenges mentioned above, we highlight the following contributions in this chapter:

- **The development of an automatic trajectory data reconciliation pipeline** for a modern traffic testbed. Specifically, the pipeline includes a) *an online data association algorithm* to solve a min-cost flow problem, which consequently matches fragments that belong to the same object, and b) a novel *trajectory reconciliation algorithm*, which is formulated as a quadratic program. It reconstructs realistic vehicle dynamics from disturbed detection data with systematic smoothing and outlier correction. The resulting trajectories automatically satisfy the internal consistency (differentiation of trajectories with speeds and accelerations).
- **The assessment on the trajectory quality including dynamics analysis, error statistics and a qualitative dashboard**. Results show that the reconciled trajectories improve a variety of measures on various traffic scenarios (simulation data, real-world freeflow and congested scenarios).
- **The deployment on I-24 MOTION system**. We build a software architecture that is capable of postprocessing upstream raw tracking data, and outputs high quality trajectory data in real time. The architecture leverages upon asynchronous processes, and is currently deployed on the I-24 MOTION system for frequent data generation.

This work, although still in progress, illustrates the first step towards automatic and online trajectory data reconciliation from video-based data extraction. Future work will focus on fine tuning the parameters associated with the proposed algorithms to further enhance the data quality.

7.2. I-24 MOTION system

7.2.1. Overview

The Interstate-24 MObility Technology Interstate Observation Network (I-24 MOTION) is a new instrument for traffic science located near Nashville, Tennessee. It consists of 276 pole-mounted high-resolution traffic cameras that provide seamless coverage of approximately 4.2 miles I-24, a 4-5 lane (each direction) freeway with frequently observed congestion. The cameras are connected via fiber optic network to a compute facility where vehicle trajectories are extracted from the video imagery using computer vision techniques. The main output of the instrument are vehicle trajectory datasets that contain the position of each vehicle on the freeway, as well as other supplementary information on vehicle dimensions and class. As the system continues to mature, all trajectory data will be made publicly available at i24motion.org/data. The map of the system can be seen in Figure 7.1.

A prototype software architecture comprises of three main modules: video ingest, vehicle detection and tracking, and trajectory post-processing and reconstruction, managed by the system control server. We refer readers our manuscript [75] for detailed descriptions of the system design, hardware and software architectures.

7.2.2. Known data artifacts

Vehicle detection and tracking is achieved through a crop-based tracking method called *Crop-based Tracking* [74]. This method uses cropped portions of images for processing, significantly reducing

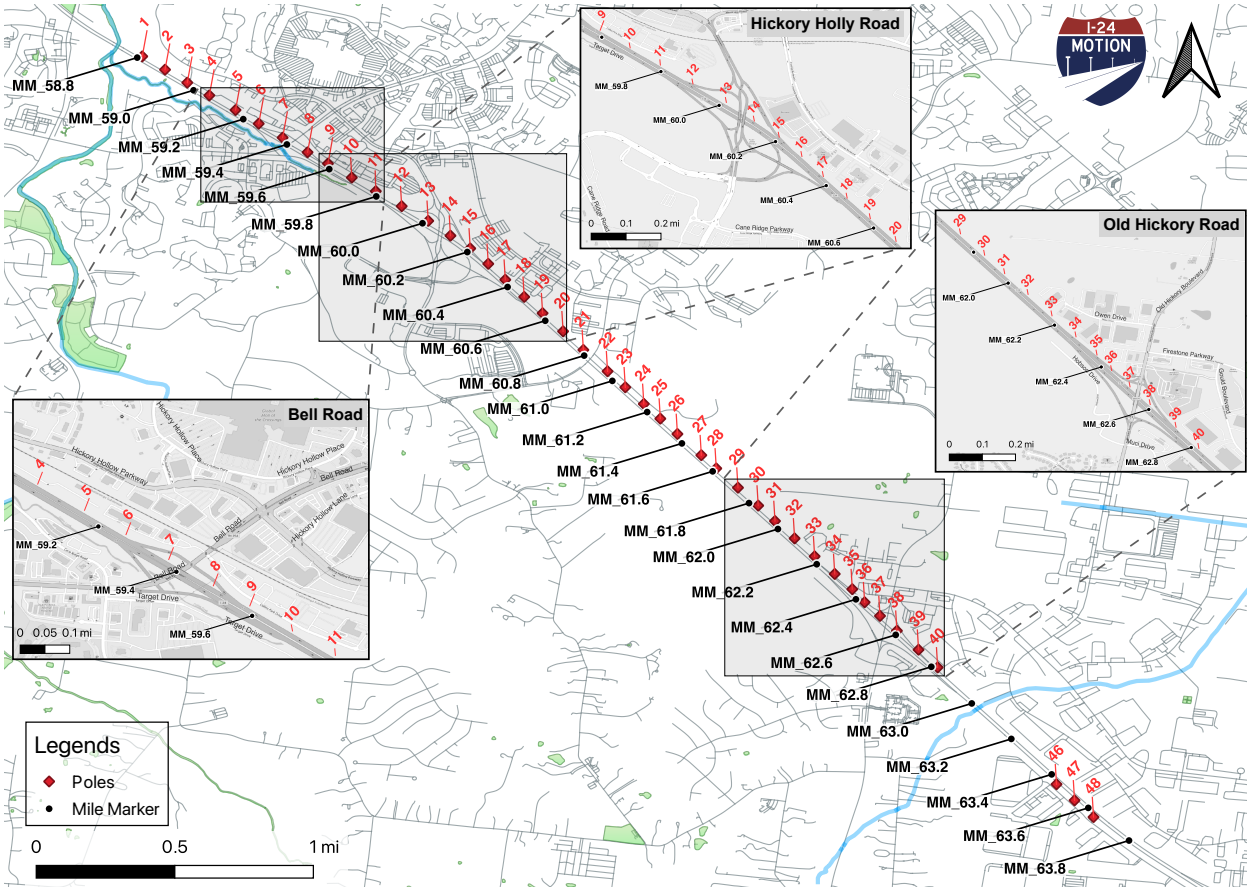


Figure 7.1: Map for I-24 MOTION infrastructure locations. Photo credit: Junyi Ji

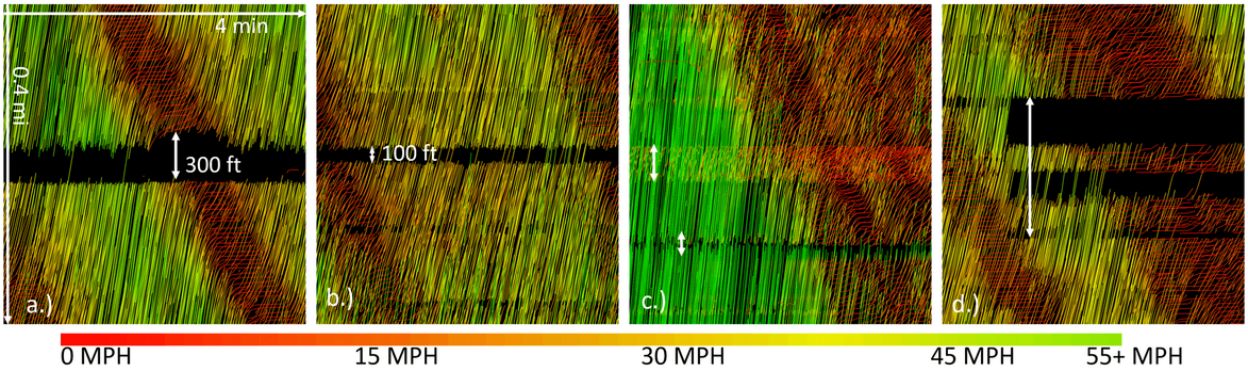


Figure 7.2: Example artifacts. For all figures, horizontal scale = 4 min. and vertical scale = 0.4 mi. a.) Missing pole / offline cameras causes a wide band of missing data. b.) Overpass causes a narrow band of missing data. c.) Homography error causes multiple trajectories corresponding to the same vehicle, or else results in a narrow band with no coverage. d.) Packet drops cause bands of missing trajectory data with a discrete start and end. Post-processing only partially fills in this data. Photo credit: Gergely Zachar and Derek Gloudemans

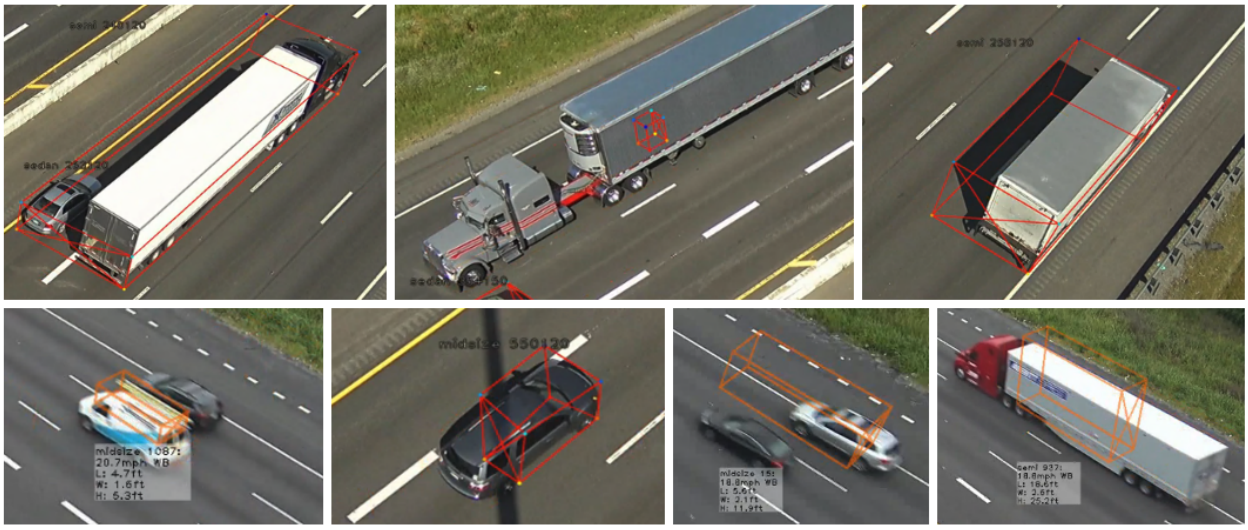


Figure 7.3: Examples of common inaccuracies from upstream video processing algorithms: missing detection, inaccurate localization, and non-rectilinear shape. Photo credit: Derek Gloudemans

detection inference time. An accurate object motion model is employed, generating cropping boxes based on object priors. A Retinanet object detector is used to detect cars and trucks. A Kalman filter with linear dynamics handles the motion model, assuming constant velocity along the primary roadway direction. Intersection-over-union metric computes object affinity. Multi-camera tracking is solved through detection fusion. The system is designed to scale with a large number of cameras, processing cameras on separate nodes.

Although raw trajectory data from dense deployment of cameras and CV algorithms can achieve complete spatial and temporal coverage of a roadway segment, such data contains inaccuracies from camera errors (dropped, doubled, and corrupted frames) network errors (Figure 7.2d: data packet drops), object detection and tracking (Figure 7.3: fragmentations, ID swaps, false negatives and false positives [22]) often caused by object-object or infrastructure-object occlusions, timestamp quantization errors, homography assumption errors, and infeasible derivative quantities resulting from finite difference approximation over very short timescales. Treatments for specific sources of errors that rely on multiple iterations of rectification or require manual fine-tuning are not viable for longer term streaming datasets the I-24 MOTION is designed to produce. For small datasets, data cleaning and rectification with some manual involvement can address many common errors created in vehicular datasets [42].

In this chapter we address the miscellaneous upstream errors in a systematic manner regardless of the causes, by introducing an automatic data post-processing pipeline which will be continuously improved to automate as much of the data cleaning steps as possible. Currently, it consists of two modules: **a) an online data association algorithm** to solve a min-cost flow problem, which consequently matches fragments that belong to the same object. This step reduces fragmentation due to tracking interruptions across multiple cameras and compute nodes, as well as potential discontinuity due to occlusion from physical infrastructure. The second component of the pipeline is **b) a trajectory reconciliation algorithm**, which is formulated as a quadratic program. This step reconstructs realistic vehicle dynamics from disturbed detection data with trajectory derivative

smoothing and outlier correction while minimally perturbing the original vehicle detections. By penalizing high-frequency noise and low-frequency outliers, this step removes the majority of the errors caused by timestamp quantization, localization errors, homography projection errors etc. The resulting trajectories automatically satisfy the internal consistency (differentiation of trajectories with speeds and accelerations).

7.3. Data reconstruction methods

7.3.1. Online data association

The multi-object tracking (MOT) problem plays a pivotal role in modern computer vision-aided cyber-physical systems. These systems rely on intelligent sensing technologies and efficient data processing tools to monitor and control physical infrastructures. However, challenges emerge when dealing with the increasing data size resulting from larger field of views covered by multiple cameras and the continuous stream of incoming data. To address these challenges, effective algorithms capable of handling significant volumes of streaming data are required. This chapter focuses on extending a well-known algorithm designed for the MOT problem into an online framework that can process streaming data of any size. The algorithm described in this chapter can effectively handle "fragments", which refer to incomplete tracking of moving objects caused by conservative tracking from the upstream video processing process. Tracking discontinuities can occur due to various factors such as object occlusion and/or misalignment between adjacent cameras.

Problem formulation

In this section, we outline the problem formulation for MOT as an equivalent problem for finding the minimum-cost circulation (MCC) of a graph. Solving for MCC on a track graph results in trajectory sets that have the highest *maximum a posteriori* (MAP). The problem formulation is explained in literature such as [264, 231], and therefore only highlighted briefly in this section.

A fragment with index k is denoted as $\phi_k = \{p_1, ..., p_n\}$, which consists of a series of positional data ordered by time (frame). Each data point p_i is a vector containing timestamp, x and y position of a fixed point on the bounding box. We are given a set of fragments as input $\Phi = \{\phi_i\}$. A trajectory $\tau_k = \{\phi_{k_1}, ..., \phi_{k_n}\}$ consists of one or more fragments. A set of such trajectories form a trajectory set hypothesis $T = \{\tau_1, ..., \tau_K\}$. Assuming that fragments are conditionally i.i.d., the fragment association step aims at finding T^*, the hypothesis with the highest MAP:

$$\begin{aligned} T^* &= \text{argmax}_T P(T|\Phi) \\ &= \text{argmax}_T P(\Phi|T)P(T) \\ &= \text{argmax}_T \prod_i P(\phi_i|T) \prod_{\tau_k \in T} P(\tau_k) \\ &\text{s.t. } \tau_k \cap \tau_l = \emptyset, \ \forall k \neq l, \end{aligned} \tag{7.1}$$

with a non-overlapping trajectory constraint, since each fragment can belong to at most one trajectory. The likelihood $P(\phi_i|T) = P(\phi_i) = \beta_i$ indicates the probability that a fragment is a false positive and thus should not be included in the trajectory hypothesis. The prior of a trajectory can

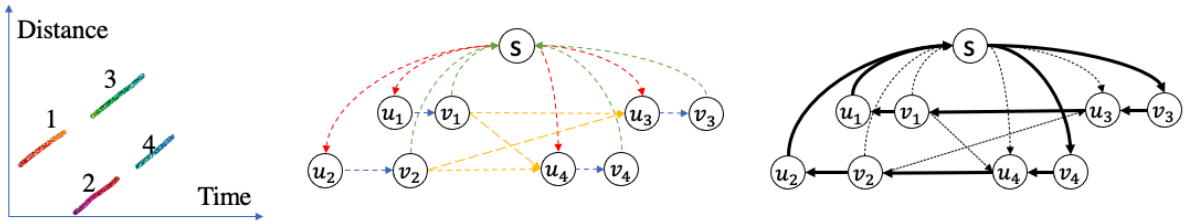


Figure 7.4: Left: fragments in time-space coordinates. In this example ϕ_1 and ϕ_3 should be associated, and ϕ_2 and ϕ_4 should be associated. The numbers indicate the order of last timestamp. Middle: fragments as a circulation graph. Red edges are the entering edges which have costs c_i^{en}; blue is inclusion edges with cost c_i; green is exiting edges with cost c_i^{ex} and yellow is the transition edges with cost c_{ij}. Right: the residual graph after running the negative cycle canceling algorithm to obtain the min-cost circulation. The residual edges that carry the min-cost circulation are highlighted in bold. The fragment association assignment can be obtained by tracing along the bold edges.

be modeled as a Markov chain:

$$P(\tau_k) = P_{\text{enter}}(\phi_{k_1}) \prod_{i=1}^{n-1} P(\phi_{k_{i+1}}|\phi_{k_i}) P_{\text{exit}}(\phi_{k_n}), \tag{7.2}$$

where $P_{\text{enter}}(\phi_{k_1})$ and $P_{\text{exit}}(\phi_{k_n})$ denote the probabilities that ϕ_{k_1} starts the trajectory and ϕ_{k_n} ends the trajectory, respectively. Taking the negative logarithm of (7.1), the MAP problem becomes equivalent to the following integer program:

$$\underset{f_i, f_{i,j}, f_i^{en}, f_i^{ex}}{\text{minimize}} \quad \sum_i c_i f_i + \sum_i c_i^{en} f_i^{en} + \sum_{i,j} c_{i,j} f_{i,j} + \sum_i c_i^{ex} f_i^{ex} \tag{7.3a}$$

$$\text{subject to} \quad f_i, f_{i,j}, f_i^{en}, f_i^{ex} \in \{0, 1\}, \tag{7.3b}$$

$$f_i^{en} + \sum_j f_{j,i} = f_i = f_i^{ex} + \sum_j f_{i,j}, \tag{7.3c}$$

where

$$\begin{aligned} c_i^{en} &= -\log P_{\text{enter}}(\phi_i), \;\; c_i^{ex} = -\log P_{\text{exit}}(\phi_i), \\ c_{i,j} &= -\log P(\phi_i|\phi_j), \;\; c_i = -\log \frac{1-\beta_i}{\beta_i}. \end{aligned} \tag{7.4}$$

The decision variables are binary according to the unit-flow constraint (7.3b). f_i indicates whether ϕ_i should be included in any trajectory, f_i^{en} and f_i^{ex} determine whether a trajectory starts or ends with ϕ_i, respectively. $f_{i,j}$ indicates if fragment ϕ_j is an immediate successor of ϕ_i. The flow-conservation constraint (7.3c) ensures non-overlapping trajectories.

Equivalent min-cost-circulation problem

In seminal work [264], it is shown that (7.3) has a natural graph interpretation, and solving for (7.3) is equivalent to solving the min-cost-flow of a tracklet graph, which has a polynomial solution [61].

Later in the work of [232], it is proven that the min-cost-flow problem for MOT is equivalent to a min-cost-circulation problem on a slightly modified graph. Many efficient algorithms are developed to solve this problem [3, 189, 109, 77], and simplification are made to further improve the algorithmic efficiency in the MOT context [264, 119, 231]. The graph is constructed such that each fragment ϕ_i is represented as two nodes u_i and v_i, with a directed edge $(u_i \rightarrow v_i)$ and a cost $\$(u_i \rightarrow v_i) = c_i$ indicating *inclusion* of ϕ_i; edges between two fragments ϕ_i and ϕ_j are represented as $(v_i \rightarrow u_j)$, with the cost $\$(v_i \rightarrow u_j) = c_{ij}$ related to the likelihood of ϕ_j following ϕ_i. The edge direction implies the sequential order between fragments. Furthermore, the graph has a dummy node s that has an incident edge to every u, and every v directs back to s. The resulting graph is therefore a directed circulation graph, see Figure 7.4. We denote this circulation graph as $G(V, E)$, with node set V and edge set E. Each edge $e := (u, v) \in E$ has a unit capacity $r(e) = 1$, a cost $\$(e)$ and a binary flow $f(e) \in \{0, 1\}$. The data association problem can be formulated as finding a set of non-overlapping circulations f on G with the lowest total cost. The total cost of the circulations is $\sum_{e \in f} \$(e) f(e)$.

Negative cycle canceling algorithm

One efficient algorithm is the negative cycle canceling algorithm (NCC) proposed by Klein [109] and later on optimized by Goldberg et al. [77, 189], based on the Ford-Fulkerson's method for incremental improvement. To understand the algorithm we first recall the definition of an important concept – a residual graph G_r:

Definition 3. The residual graph $G_r(V, E_r)$ for the original directed graph $G(V, E)$ with respect to a flow f is generated by replacing each edge $e = (u \rightarrow v) \in E$ by two residual edges $e' = (u \rightarrow v) \in E_r$ and $e_r = (v \rightarrow u) \in E_r$, with cost $\$(e') = \$(e)$ and residual capacity $r(e') = r(e) - f(e)$, while $\$(e_r) = -\$(e)$ and $r(e_r) = f(e)$.

In the context of MOT graph as shown in Figure 7.4, the construction of residual graph can be simplified. The edges in the flow of the original graph simply needs to be reversed and costs on the edges negated, to form the corresponding residual graph.

The idea of NCC is to repeatedly find a cycle with negative cost in the residual graph G_r and push flow through the cycles. The algorithm terminates when no more negative cycles can be found (optimality condition). We direct interested readers the above reference for the details and proof of correctness of this algorithm, and only provide an outline in Algorithm 5.

First, a circulation graph $G(V, E)$ is constructed from the set of fragments Φ (ConstructTracklet-Graph) and we iteratively look for a negative cycle in G_r based on, for example, Bellman-Ford algorithm. If such cycle exists, then update the residual graph according to Definition 3 (Push-Flow). When the iteration stops (no more negative cycle can be found), the assignment, or the trajectories, can be extracted by traversing along all the cycles through the residual edges in G_r (FlowToTrajectories).

Note that the NCC algorithm guarantees feasibility at each iteration because every node is balanced (inflow equals outflow). The algorithm terminates immediately if no feasible flow can be found. The proof for correctness of the general NCC algorithm is detailed in [3]. The central idea is based on the theorem of negative cycle optimality conditions:

Theorem 2. (Negative Cycle Optimality Conditions) A feasible circulation f in G is optimal if and only if the residual graph G_r contains no negative-cost cycles.

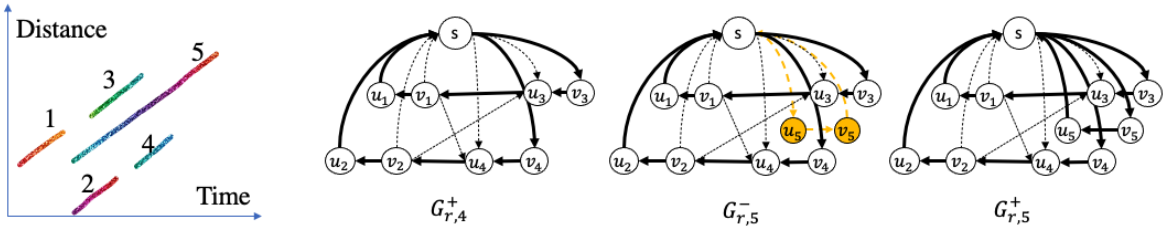


Figure 7.5: Scenario 1: the new fragment (ϕ_5) starts a new trajectory. The residual graph from the previous iteration is $G^+_{r,4}$, with the min-cost circulation highlighed in bold. $G^-_{r,5}$ is obtained by AddNode($G^+_{r,4}$, ϕ_5), with the added nodes highlighted in yellow. The min-cost cycle Γ in $G^-_{r,5}$ is colored yellow. Finally, $G^+_{r,5}$ is obtained by PushFlow($G^-_{r,5}$, Γ).

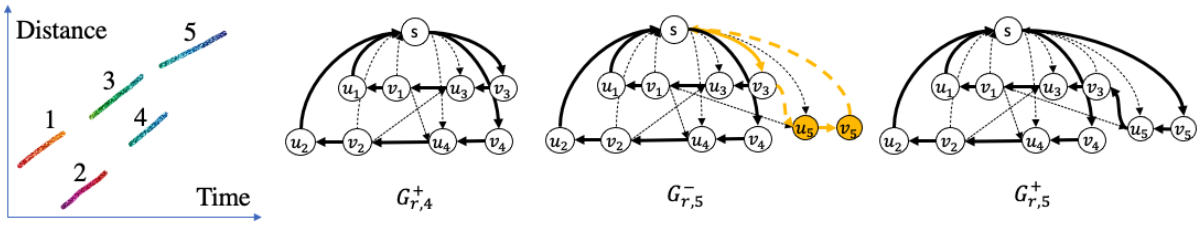


Figure 7.6: Scenario 2: the new fragment is connected to the tail of an existing trajectory. In this example the new node u_5 has candidate connections to v_1 and v_3 based on the motion model described in [246]. The min-cost cycle in this scenario also includes the post-node of ϕ_3, v_3, which means that ϕ_5 succeeds ϕ_3 as the new tail of this trajectory.

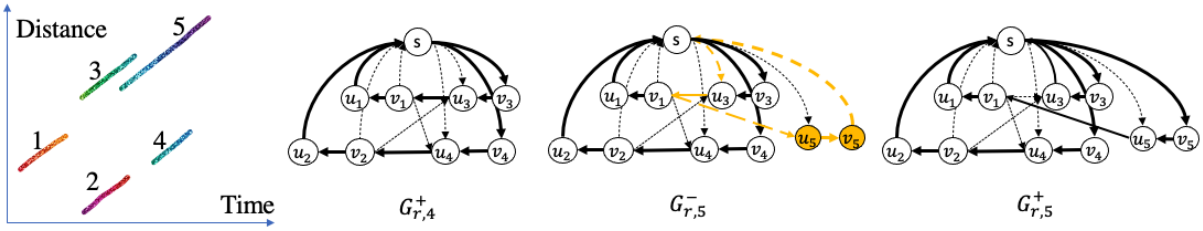


Figure 7.7: Scenario 3: the new fragment breaks an existing trajectory. In this case, the min-cost cycle contains the pre-node of ϕ_3 and the post-node of ϕ_1, meaning ϕ_5 has a higher tendency to be a continuation of ϕ_1 and ϕ_3 is siloed.

Additionally we mention the following useful lemma that is specific to the MOT context, and was proved in [232]:

Lemma 1. A circulation with total integer flow amount K can only be sent through K distinct cycles.

Next we show an online extension of the NCC algorithm and provide proof for correctness based on this important theorem.

Algorithm 5 Negative cycle cancellation for min-cost-circulation on a tracklet graph

Input: Set of fragments $\Phi = \{\phi_i\}$
Result: Set of trajectories $T = \{\tau_i\}$
  $G(V, E, C) \leftarrow$ ConstructCirculationGraph(Φ)
  $f \leftarrow 0$
  $G_r \leftarrow G$
while *a negative-cost cycle* Γ *exists in* G_r **do**
  // Update residual graph
  $G_r \leftarrow$ PushFlow(G_r, Γ)
end
$T \leftarrow$ FlowToTrajectories(G_r)

Online negative cycle canceling

The streaming data coming from modern sensing technologies necessitates an online and memory-bounded version of Algorithm 5. In other words, the tracking graph G is dynamic: new fragments are added in order of time and older fragments are removed from the graph. In this section we introduce an online version of the NCC algorithm which can be applied to a dynamic graph for online MOT.

A naive online extension of Algorithm 5 is to construct a circulation graph for each newly added fragment ϕ_k from scratch and rerun the NCC algorithm. However, it is inefficient because the majority of the graph remains the same and the majority of the computation on the min-cost cycle is wasted. This opens opportunities for an online extension of the algorithm to minimize repeated calculations.

The proposed online algorithm is based on the assumption that fragments are added to the graph in the order of last timestamp, which is a reasonable assumption in practice as this is the order that fragments are generated from object tracking. The online algorithm proceeds by adding each fragment ϕ_k to the residual graph from the previous iteration $G^+_{r,k-1}$ one at a time, to obtain a new graph $G^-_{r,k}$ (AddNode($G^-_{r,k-1}, \phi_k$)). This step adds two nodes u_k and v_k to the graph along with edges $(s \to u_k), (u_k \to v_k), (v_k \to s)$ and possibly additional transition edges incident to u_k. Then, we search for the least-cost negative cycle Γ in $G^-_{r,k}$ (FindMinCycle($G^-_{r,k}$)) and push flow through the cycle to obtain the updated residual graph $G^+_{r,k}$. When all the fragments are processed, we output the trajectories T by tracing all the cycles in the final residual graph. It can be proved that pushing flow through Γ, $G^+_{r,k}$ contains the min-cost circulation because the flow is feasible and no further negative cycles can be found in G^+_r. We denote the residual graph after adding ϕ_k at iteration k to be $G^+_{r,k}$. The algorithm is shown in Algorithm 6.

A visual illustration of how the online NCC algorithm works is shown in Figure 7.5-7.7. We show three scenarios for adding a new fragment ϕ_5 to the current MOT result with 4 fragments (The MOT result up to $k = 4$ is maintained by $G^+_{r,4}$, where ϕ_1 and ϕ_3 are temporarily associated, and ϕ_2 and ϕ_4 are temporarily associated).

Algorithm 6 Online NCC for MCC on a tracklet graph

Input: Set of fragments $\Phi = \{\phi_i\}$
Result: Set of trajectories $T = \{\tau_i\}$
  $f \leftarrow 0$
  $G^+_{r,0} \leftarrow \{s\}$
  $k \leftarrow 1$
for *each* ϕ_k *(ordered by last timestamp)* **do**
  $G^-_{r,k} \leftarrow \text{AddNode}(G^+_{r,k-1}, \phi_k)$
  $\Gamma \leftarrow \text{FindMinCycle}(G^-_{r,k})$
  $G^+_{r,k} \leftarrow \text{PushFlow}(G^-_{r,k}, \Gamma)$
  $k \leftarrow k + 1$
end
$T \leftarrow \text{FlowToTrajectories}(G^+_{r,k})$

The proof for correctness of Algorithm 6 is described in Appendix C, along with an improvement on runtime and memory. Algorithm 7 documented in the Appendix is currently deployed on the I-24 MOTION system.

7.3.2. **Trajectory reconciliation algorithm**

Problem formulation

After applying fragment association, the next step is to rectify the stitched, raw trajectories with denoising, imputation and smoothing operations. Instead of ad-hoc correcting each source of the detection errors shown in Figure 7.3, we treat all noises and errors in a one-step approach to rectify them all at once. To simplify the problem, we consider a 2D vehicle motion model, with independent longitudinal (x) and lateral (y) dynamics. This allows us to decompose the problem to solving two independent 1D reconciliation problems. The reconciled trajectories automatically satisfy the state consistency amongst the orders of differentiation, i.e., the finite-difference of position is speed and the finite-difference of speed is acceleration, etc.

We use the following notations. Each trajectory τ_i includes the following features (features are the same for all trajectories, therefore the index i is dropped for simplicity): let $\mathbf{x} = [x[1], x[2], ..., x[N]]^T$ be the back-center x-position vector for N timesteps, and similarly let $\mathbf{y}$ be the time-series of the back-center y-position; $\mathbf{v}_x = [v_x[1], v_x[2], ..., v_x[N-1]]^T$ is the time-series of speed in the longitudinal direction (x-axis), and similarly $\mathbf{v}_y$ is the time-series of speed in the lateral component (y-axis). The acceleration $\mathbf{a}$ and jerk $\mathbf{j}$ use the same 2D representation. Lastly, l, w represent the vehicle length and width, respectively. The vehicle dynamics model can be seen in Figure 7.8

Consider the following discrete-time 3rd order 1D motion model in either longitudinal or lateral direction (the same method is applied for x-component and y-component dynamics. We demonstrate

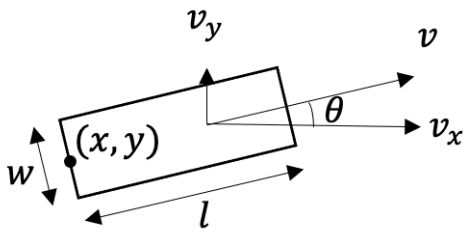


Figure 7.8: A simple 2D vehicle model

on x-component only):

$$\begin{aligned} x[t+1] &= x[t] + v[t]\Delta T \\ v[t+1] &= v[t] + a[t]\Delta T \\ a[t+1] &= a[t] + j[t]\Delta T, \end{aligned} \tag{7.5}$$

with ΔT as the timestep. Notice that the finite-difference method decrements the dimension of time-series as an increase of derivative order, i.e., $\mathbf{x} \in \mathbb{R}^N$, $\mathbf{v} \in \mathbb{R}^{N-1}$, $\mathbf{a} \in \mathbb{R}^{N-2}$ and $\mathbf{j} \in \mathbb{R}^{N-3}$. Eq (7.5) can be written in matrix multiplication form:

$$\begin{aligned} \mathbf{v} &= D^{(1)}\mathbf{x} \\ \mathbf{a} &= D^{(2)}\mathbf{x} \\ \mathbf{j} &= D^{(3)}\mathbf{x}, \end{aligned} \tag{7.6}$$

where $D^{(k)} \in \mathbb{R}^{(N-k)\times N}$ represents the k^{th}-order differentiation operator. For example, $k = 1, 2,$ and 3 can be written as:

$$D^{(1)} = \frac{1}{\Delta T}\begin{bmatrix} -1 & 1 & 0 & ... & 0 & 0 & 0 \\ 0 & -1 & 1 & ... & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & ... & -1 & 1 & 0 \\ 0 & 0 & 0 & ... & 0 & -1 & 1 \end{bmatrix}$$

.

$$D^{(2)} = \frac{1}{\Delta T^2}\begin{bmatrix} 1 & -2 & 1 & 0 & ... & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & ... & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & .... & 1 & -2 & 1 \end{bmatrix}$$

$$D^{(3)} = \frac{1}{\Delta T^3}\begin{bmatrix} -1 & 3 & -3 & 1 & ... & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & .... & -1 & 3 & 3 & 1 \end{bmatrix}.$$

Consider corrupted measurement $\mathbf{z}$ which contains missing data (indicated by the observation operator H), noises $\mathbf{w}$ and outliers $\mathbf{e}$:

$$\mathbf{z} = H\mathbf{x} + \mathbf{w} + \mathbf{e}, \quad \mathbf{z} \in \mathbb{R}^M, \mathbf{w} \in \mathbb{R}^M, \mathbf{e} \in \mathbb{R}^M, H \in \mathbb{R}^{M\times N}, \tag{7.7}$$

and missing data exists if $M < N$. We aim to find the reconstructed position $\hat{\mathbf{x}}$ that is smooth in k^{th}-order derivatives. The idea is to use a combination of Ridge and Lasso regression [268], to simultaneously handle noises and outliers, assuming outliers are sparse and noises have small magnitude:

$$\underset{\mathbf{x},\,\mathbf{e}}{\text{minimize}} \quad \|\mathbf{z} - H\mathbf{x} - \mathbf{e}\|_2^2 + \sum_{k=2}^{k=K} \lambda_k \|D^{(k)}\mathbf{x}\|_2^2 + \lambda_1 \|\mathbf{e}\|_1 \tag{7.8a}$$

$$\text{subject to} \quad -D^{(1)}\mathbf{x} \preceq 0, \tag{7.8b}$$

$$\lfloor \mathbf{x} \rfloor^{(k)} \preceq D^{(k)}\mathbf{x} \preceq \lceil \mathbf{x} \rceil^{(k)}, k = 2, 3, ..., K. \tag{7.8c}$$

The first term of the cost function (7.8a) penalizes the data-fitting error on the non-missing entries. The second term regularizes the smoothness of the position vector, by penalizing the l_2-norm of higher-order derivatives (e.g., $k = 2$ and $k = 3$ correspond to accelerations and jerks in respective). The third term regularizes the sparsity of the outliers. The first constraint (7.8b) states that the speed has to be non-negative, i.e., no cars are traveling backward at any time. The second constraint (7.8c) sets the upper and lower bound for each high-order derivatives. For example, $\lceil \mathbf{x} \rceil^{(2)}$ is the largest possible acceleration. Note that (7.8a) can be written into a quadratic programming form by converting the l_1 penalization to a linear programming with linear inequality constraints [36] [28]. The problem can be solved with a convex programming solver such as `cvxopt` [216]. Note that above formulation rectifies the trajectory of each vehicle independently of another. After treatment (7.8), further investigation is needed to determine if, for example, considering vehicular interactions in the optimization formulation (e.g., with additional non-collision constraint) is required.

The 2D dynamics can be obtained by solving two independent optimization problems of form (7.8) for both longitudinal and lateral dynamics.

7.4. Experiment and deployment on the I-24 MOTION system

In this section we discuss the application of the proposed two-step data postprocessing pipeline on the newly established I-24 MOTION system. A comprehensive evaluation is conducted on the validation system, a small subset of the I-24 MOTION with 18 cameras, where a short segment of manually-labeled ground truth trajectory data is used as benchmark. More qualitatively (in the absence of ground truth), we computed a variety of traffic-specific measures such as speed and acceleration distributions, to evaluate the truthfulness of the resulting postprocessed data.

7.4.1. Experiments on the I-24 MOTION validation system

The validation system was built in 2020 and functioned as a prototype for the design selections in the full system. The validation system consists of three poles that host 18 cameras to seamlessly cover 2000 ft highway segment. The detailed description of the design can be found in [75].

Evaluation metrics

Vehicles tracking and trajectory reconciliation performance can be evaluated with standard *multi-object-tracking* (MOT) metrics, specified in [22, 145, 125, 180]. For all the following metrics, an intersection-over-union (IOU) of vehicle footprint with ground truth of 0.3 or higher is required to be considered as a true positive.

- *Precision*: number of detected objects over sum of detected and false positives (target: 1).
- *Recall*: number of detections over number of objects (target: 1).
- *Switches per GT*: total number of track switches per ground truth trajectory (target: 0).
- *Fragments per GT*: total number of switches from tracked to not tracked per ground truth trajectory (target: 0).
- *Multi-object-tracking accuracy* (MOTA): an aggregated measure to indicate tracking performance. It is detection errors (false negatives and false positives) and fragmentations normalized by the total number of true detections (target: 1).
- *Multi-object-tracking precision* (MOTP): the total error in estimated position for matched prediction and ground truth pairs over all time, averaged by the total number of matches made (target: 1).

Other statistics are computed to qualitatively assess the trajectories:

- Trajectory lengths distribution
- Speed distribution (calculated using finite-difference)
- Acceleration distribution (calculated using finite-difference)

Dataset description

Properties	GT-i	GT-ii
Length of video recording (sec)	90	51
Length of road segment (ft)	2073	2073
Total number of vehicle annotations	176,805	91,267
No. lanes	8	8
No. trajectories	314	100
Trajectory lengths (ft) (avg, stdev, max)	(1635.4, 639.0, 2270.5)	(1013.5, 587.9, 2183.8)
Speed (ft/sec) (mean, stdev, max)	(106.0, 10.9, 147.7)	(42.2, 35.1, 110.8)

Table 7.1: Ground Truth Dataset Statistics

All evaluations are performed on two internal datasets obtained from the I-24 MOTION validation system, to be subsequently released to the public. The first dataset consists of 176,805 vehicle annotations spanning 90 seconds of video capturing free-flow traffic from each of 18 traffic cameras, with ground truth vehicle bounding boxes obtained by human annotators for all vehicles within the video sequences. Similarly, the second dataset contains 91,267 annotations from a 51-second recorded video that capture denser and slower traffic than the one from the first recording. The two ground truth datasets are termed as GT-i and GT-ii, respectively. Table 7.1 summarizes the two ground truth datasets in more detail.

The raw tracking data was obtained using a crop-based fast tracking algorithm mentioned in [74]. The raw tracking results obtained from the two video recordings are termed as RAW-i and RAW-ii. Likewise, the postprocessed datasets are denoted as REC-i and REC-ii in respective.

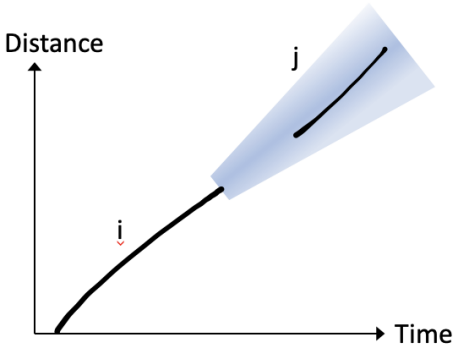


Figure 7.9: Probability of fragment ϕ_i after the last measurement of ϕ_i is represented as a cone. The matching cost of ϕ_i and ϕ_j is determined by the negative log likelihood of ϕ_j given the cone probability.

Implementation details

In this section, we describe the computation of edge costs for min-cost flow and the choice of parameters λ_i's for solving the trajectory rectification problem 7.8.

The probabilities of a trajectory starts or ends with a fragment P_{enter} and P_{exit} is determined by the ratio of the true number of trajectories and the predicted number of trajectories. The fragment linking costs $P(\phi_i|\phi_j)$ can be modeled by (a combination of) dynamics, shape affinity and the time interval between the two fragments. We implemented a version that considers the dynamics only, as it is shown to properly represent the matching probabilities.

The general idea is, if ϕ_j is indeed a continuation of ϕ_i but the object tracking is broken due to, e.g., object occlusion, then the projected position of ϕ_i at the time of ϕ_j has a high chance of overlapping with the actual detection of ϕ_j. The probability of track i at the time of ϕ_j follows a stochastic process, with the mean $\hat{p}_i$ being the projected position of track i given ϕ_i, and the variance increases with respect to Δt, the time elapsed since the end of ϕ_i, t_e^i, i.e.,

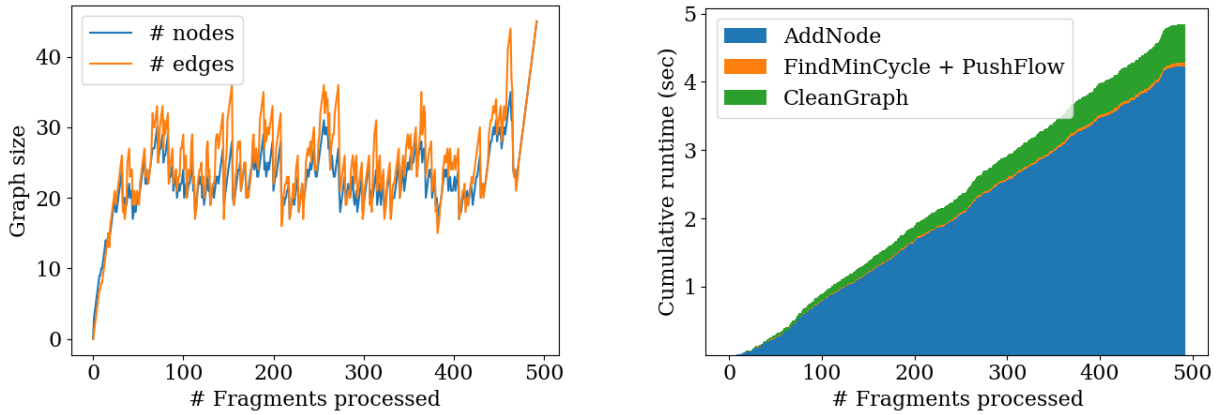
$$p_i(t_e^i + \Delta t) = \hat{p}_i(t_e^i + \Delta t) + \eta_i(\Delta t), \text{ and } \eta_i(\Delta t) \sim \mathcal{N}(0, \alpha + \beta \Delta t). \tag{7.9}$$

The uncertainty $\eta_i(\Delta t)$ follows a Brownian-like process, with zero-mean and variance $\alpha + \beta\Delta t$ growing linearly with Δt (see Figure 7.9). The projected mean of position is:

$$\hat{p}_i(t) = v_i t + \bar{p}_i, \tag{7.10}$$

where $v_i = [v_{x,i}, v_{y,i}]^T$ and $\bar{p}_i = [x_i, y_i]^T$ can be determined using, for example, linear regression. Finally, the matching cost is the negative log likelihood of ϕ_j given the probabilities computed from ϕ_i:

$$\Lambda(\phi_i, \phi_j) = \frac{1}{2N_j} \sum_{t_j \in [t_s^j, t_e^j]} \log\left(\alpha + \beta(t_j - t_e^i)\right) + \frac{1}{2N_j} \sum_{t_j \in [t_s^j, t_e^j]} \frac{(p_j(t_j) - \hat{p}_i(t_j))^2}{\alpha + \beta(t_j - t_e^i)}, \tag{7.11}$$



(a) Graph size at each iteration of online NCC.

(b) Cumulative runtime (sec) for online NCC.

Figure 7.10: Online NCC performance

where N_j is the number of measurements of ϕ_j. t_s^j and t_e^j are the start and end timestamp of ϕ_j, and $p_j(t_j)$ is the measurement of ϕ_j at t_j.

After fragment association, the next step is to impute missing data, correct for outliers, and smooth the trajectories in a single step by solving an optimization problem (Eq. (7.8)). To balance the terms in the cost function, we conducted a grid search to determine the optimal parameters λ_i. We found that setting K up to 3, where the λ_2 term penalizes large accelerations and λ_3 penalizes large jerks, was sufficient. Empirical assessment of a subset of trajectories led us to pre-tune the parameters to $\lambda_1 = 1.0 \times 10^{-3}$, $\lambda_2 = 1.67 \times 10^{-2}$, and $\lambda_3 = 1.0 \times 10^{-7}$, for both the longitudinal and lateral directions of travel. The acceleration and jerk constraints were set to ± 10 ft/sec^2 and ± 3 ft/sec^3 for the longitudinal (x) direction, and ± 5 ft/sec^2 and ± 1 ft/sec^3 for the lateral direction, respectively.

The parameter tuning is subject to the implicit assumption of Euler forward discretization, with the current sampling frequency of 25Hz. We demonstrate the capacity the above choice of parameters to bring close the speed and acceleration to a more realistic range, but note that a detailed parameter optimization that generalizes to a range of datasets is part of our ongoing effort to operationalize the described methods.

Results

The performance of the pipeline on two tracking datasets is presented in Table 7.2 and 7.3. The results demonstrate that the data association step enhances precision and recall by accurately connecting fragments and filling in the gaps between the detections.

The graph size remains bounded during the online NCC procedure. We choose a window size of 5 seconds, meaning that when adding fragment ϕ_k with a last timestamp of t_k, all existing trajectories with a last timestamp older than $t_k - 5$ are removed from the graph using the CleanGraph step. Figure 7.10a illustrates the relationship between the graph size and the number of fragments processed k. We observe that the number of nodes is maintained between 20-30, and the number

Metrics / Statistics	RAW-i	REC-i
Precision ↑	0.71	0.90 (+26.8%)
Recall ↑	0.56	0.83 (+48.2%)
MOTA ↑	0.32	0.74 (+131.3%)
MOTP ↑	0.63	0.73 (+15.9%)
Fragments per GT ↓	5.22	0.60 (-88.5%)
Switches per GT ↓	1.43	0.04 (-97.2%)
No. unique trajectories	789	321
Trajectory lengths (ft) (avg, stdev, max)	(507.8, 419.4, 2096.5)	(1455.1, 711.6, 2094.7)
Speed (ft/sec) (avg, stdev, max)	(106.4, 13.8, 437.8)	(105.9, 10.5, 142.9)
Acceleration (ft/sec^2) (avg, stdev, max)	(-9.76, 1891.3, 55701.0)	(0.12, 1.46, 6.7)

Table 7.2: Evaluation results for the first ground truth dataset.

Metrics / Statistics	RAW-ii	REC-ii
Precision ↑	0.87	0.88 (+1.1%)
Recall ↑	0.55	0.79 (+43.6%)
MOTA ↑	0.48	0.68 (+41.7%)
MOTP ↑	0.72	0.75 (+4.2%)
Fragments per GT ↓	5.38	1.95 (-63.8%)
Switches per GT ↓	2.98	0.53 (-82.2%)
No. unique trajectories	411	150
Trajectory lengths (ft) (mean, stdev, max)	(147.8, 126.8, 568.8)	(588.2, 512.2, 2181.6)
Speed (ft/sec) (mean, stdev, max)	(39.2, 11.6, 276.5)	(39.3, 7.4, 66.4)
Acceleration (ft/sec^2) (mean, stdev, max)	(39.2, 675.8, 25017.1)	(-0.04, 1.48, 6.94)

Table 7.3: Evaluation results for the second ground truth dataset.

of edges between 20-35, ensuring it remains “memory-bounded". The graph size decreases each time when timed-out trajectories (nodes) are removed from the graph. Towards the end of the iteration, the size temporarily increases because fewer nodes are removed as the time cursor no longer advances forward.

The cumulative run-time for each process in the online NCC algorithm is shown in Figure 7.10b. The total run-time of this example is approximately 5 seconds, averaging about 0.01 seconds per iteration. This time is well below the input rate and scales linearly with the number of fragments in the dataset. In this example, the majority of the run-time is consumed during the AddNode step, where the cost of every pair of fragments in the time window needs to be computed. Notably, the runtime of FindMinCycle and PushFlow combined constitutes only 1% of the total computation time at each iteration (10^{-4}sec per iteration), which is the optimization target of the online NCC algorithm.

Furthermore, the smoothing step leads to a more realistic range of speed and acceleration, as compared to the raw tracking data which contains quantization errors. Figure 7.11 and 7.12 provide a comparison of trajectory lengths, speed, and acceleration distributions before and after postpro-

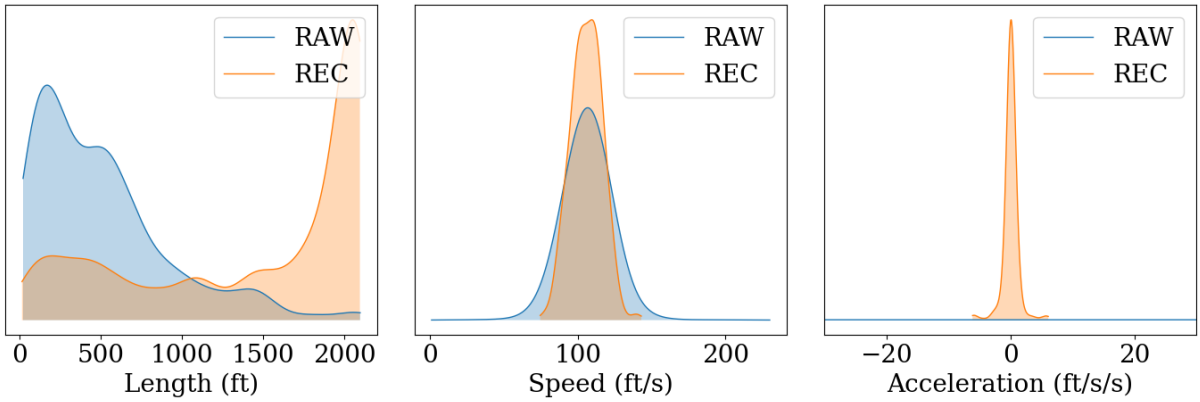


Figure 7.11: Distributions of trajectory lengths (left), speed (middle) and acceleration (right) for RAW-i (blue) and REC-i (orange).

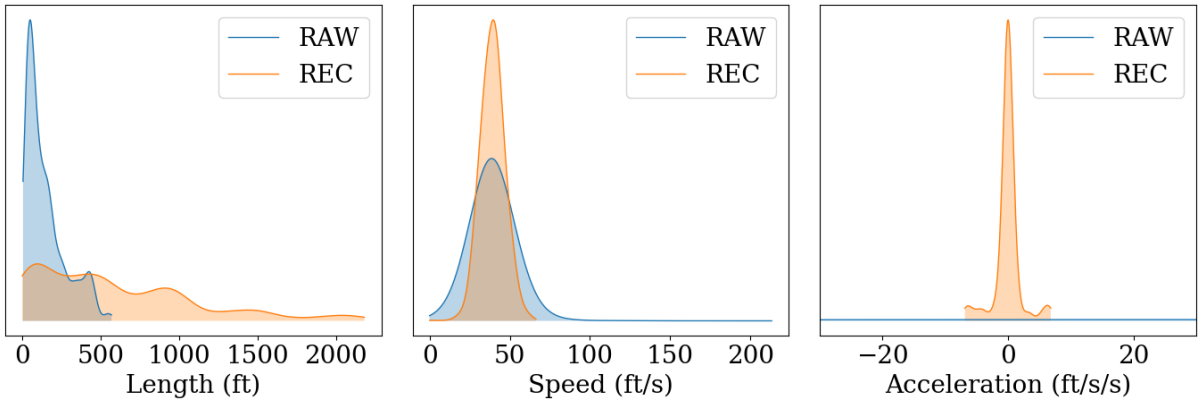


Figure 7.12: Distributions of trajectory lengths (left), speed (middle) and acceleration (right) for RAW-ii (blue) and REC-ii (orange).

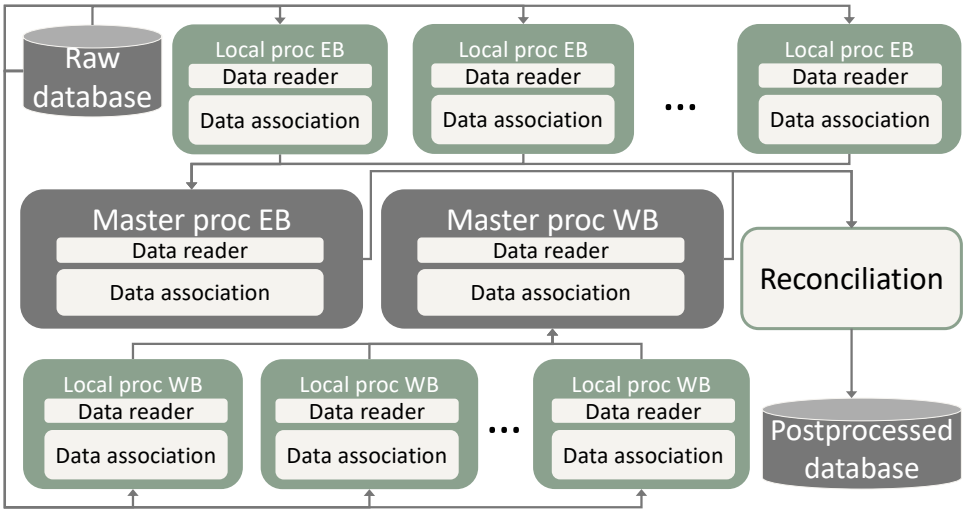


Figure 7.13: An overview of the postprocessing system diagram. The architecture includes several parallel processes performing local data association in each direction (EB: east bound, WB: west bound). The results are then passed on to the master processes, which perform association across adjacent road segments. The final step involves the reconciliation module, which imputes and smooths all associated fragments and writes the results to the database.

cessing.

In summary, the postprocessing improves all metrics on all the tested datasets. Particularly it produces longer trajectories with feasible dynamics. However, the most significant challenge occurs when associating fragments in dense and slow traffic, where the complex speed and lane-change behavior is not adequately captured by the current data association cost model.

7.4.2. **Deployment on I-24 MOTION System**

The complete I-24 MOTION testbed [75] consists of 276 cameras in total that seamlessly cover 8 lanes (4.2 miles) of I-24 segment near Nashville, TN (see the testbed map in Figure 7.1). On a typical workday morning, the recorded raw tracking data spans 4 hours, from 6:00 AM to 10:00 AM, making it 16 times larger in temporal scale, and 22 times larger in spatial scale than SIM3, the largest simulation dataset presented in this chapter. Therefore, it is crucial to design an architecture that is scalable for size of the testbed.

Software architecture

To handle the volume of tracking data from this testbed, we design a postprocessing software architecture that employs parallel and asynchronous compute processes (see Figure 7.13). This architecture utilizes multiple processes running concurrently, managed by a master scheduler, to handle tracking data from nine upstream video processing nodes, with each process assigned to independently process fragments detected from 25-31 cameras. Each process performs local data association independently, and the locally processed results are passed to the master process of the corresponding direction of travel, which runs a second pass of data association to connect the partial trajectory fragments across each adjacent local process. Finally, all resulting trajectories are smoothed and imputed in the "reconciliation module" before being written to the postprocessed database.

Our experiments demonstrate that this software architecture can effectively handle the volume of tracking data from this testbed. In addition to implementing an online version of the min cost flow algorithm mentioned in Section 7.3.1, for a typical workday morning on I-24 MOTION, which spans 8 lanes of traffic (4.2 miles) and was captured from 6AM to 10AM, our approach can process approximately 400 trajectories per minute for light traffic (flow is about 30 vehicles/lane/min), while for heavy traffic (120 vehicles/lane/min) it can process one minute of data in 50 seconds. For a full-scale 4-hour run with the asynchronous software structure illustrated in Figure 7.13, local processes are completed in 1.5 hours, and master processes in 2 hours, resulting in a total runtime of 3.5 hours to process 4 hours of data. The server that hosts this architecture operates on Ubuntu 20.04, with 528GB total memory and 64 CPU cores. These results indicate that our proposed algorithm is scalable and can efficiently process large volumes of real-time tracking data on I-24 MOTION.

Time-space diagrams

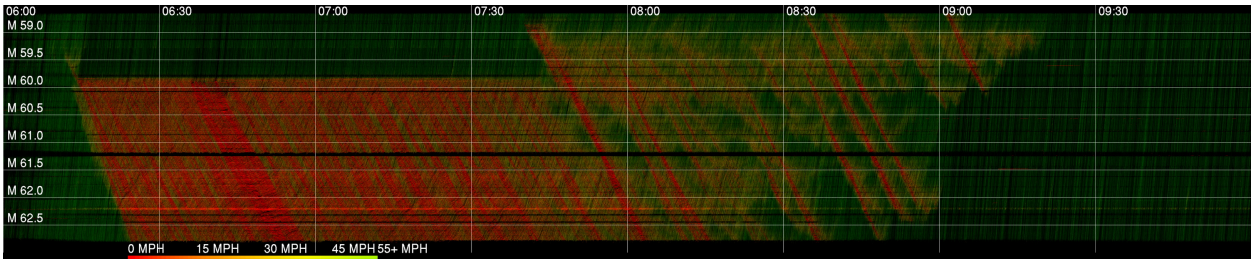
Figures 7.14a-7.14e illustrate a subset of data captured by I-24 MOTION. Each day, the traffic is shown as a time-space diagram spanning 4.2 miles of I-24 westbound traffic during 4 hours of morning congestion starting at 6:00AM. Each image is created by plotting all westbound vehicle trajectories and color-coding the points based on the speed of the vehicle. Vehicle lengths, widths, heights, and lateral positions are also measured but not shown. The waves visible in the image propagate at approximately 12-13 miles per hour.

The time-space diagrams for I-24 W morning rush hour traffic from Nov 21 - Nov 25, 2022 are read such that the x-axis is time of day (HH:MM), and y-axis is roadway postmile (mi). Postmile decreases for travelers in the westbound direction. A typical congestion pattern is shown with frequent oscillatory traffic observed. In this week, several incidents were observed. For example, on Nov 21 (Figure 7.14a) A severe rear-end crash on the HOV lane occurred at 6:14AM that was immediately followed by an onset of upstream queuing on lane 1 and lane 2. The congestion lasted for about 1.5 hrs before the crash was cleared. Later at 7:40AM, there was a slowdown on lane 3 caused by a large object falling out of a pickup truck. The roadway was cleared about 2.5 minutes later. On Nov 23 (Figure 7.14c), the traffic volume reduced due to the approach of the Thanksgiving holiday. A sideswipe crash occurred at 7:35AM due to a vehicle changing from lane 1 to lane 2 that caused a collision with another car travelling in lane 2. All of these events result in noticeable congestion patterns shown in the time-space diagrams.

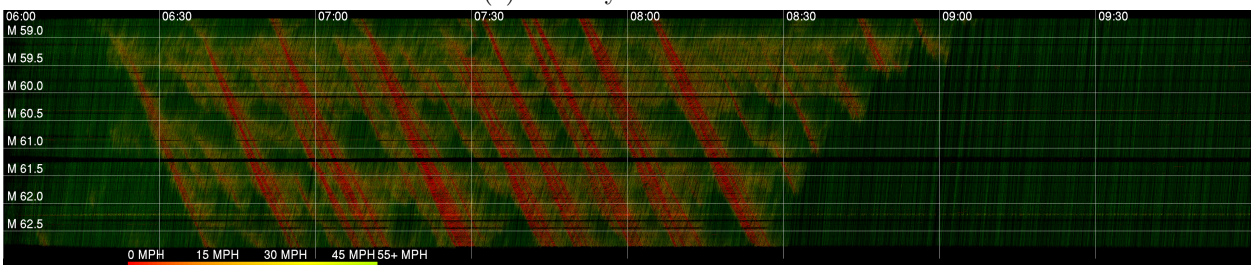
The diagrams show that perturbations in different times and locations all propagate upstream,

creating traffic waves with varying frequencies and amplitudes. Although the severity of the bottlenecks vary, the resulting congestion pattern generally travel against the direction of traffic at a constant characteristic speed of approximately 13 mph (see also [209, 87, 104]). We observe that oscillations with longer periods are often accompanied by larger amplitudes. The strong correlation between traffic wave period and amplitude is also discussed in [64].

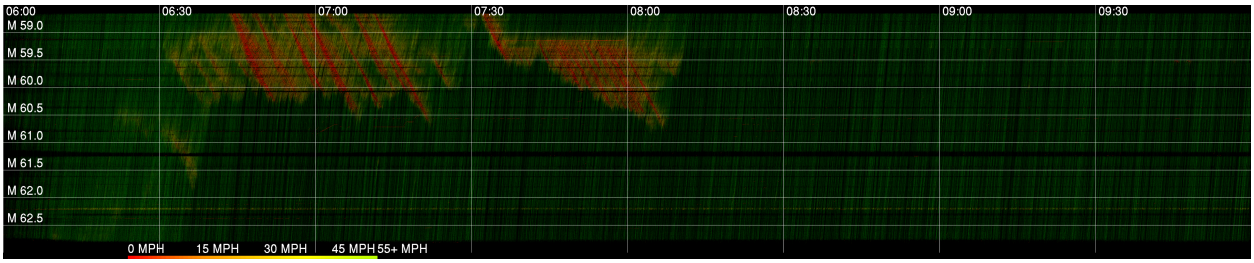
Even in the present form, data from I-24 MOTION already suitable to study traffic waves and other macroscopic quantities. This allows I-24 MOTION data to be used for speed analysis directly without needing to extrapolate long distances between fixed sensors (data cleaning is, however still required). Moreover, the camera-based sensors yield useful insight into the initial causes of bottlenecks not visible in any other sensing modality (e.g., debris on the roadway).



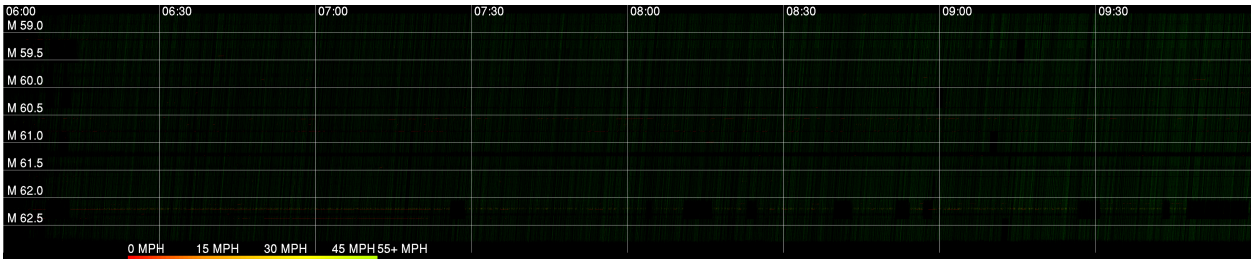
(a) Monday Nov 21 2022



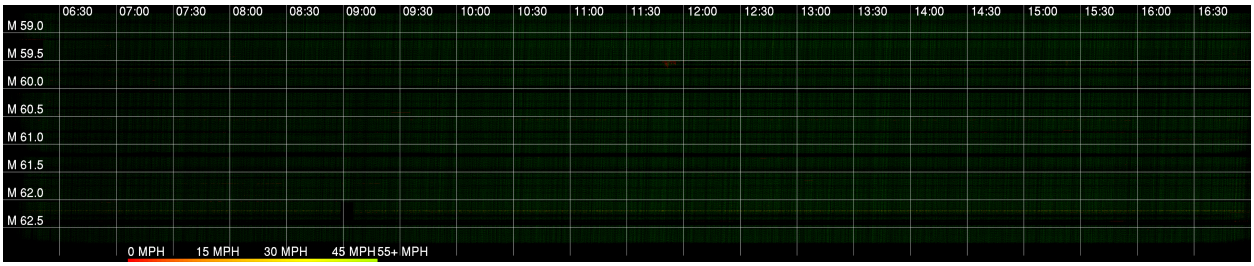
(b) Tuesday Nov 22 2022



(c) Wednesday Nov 23 2022

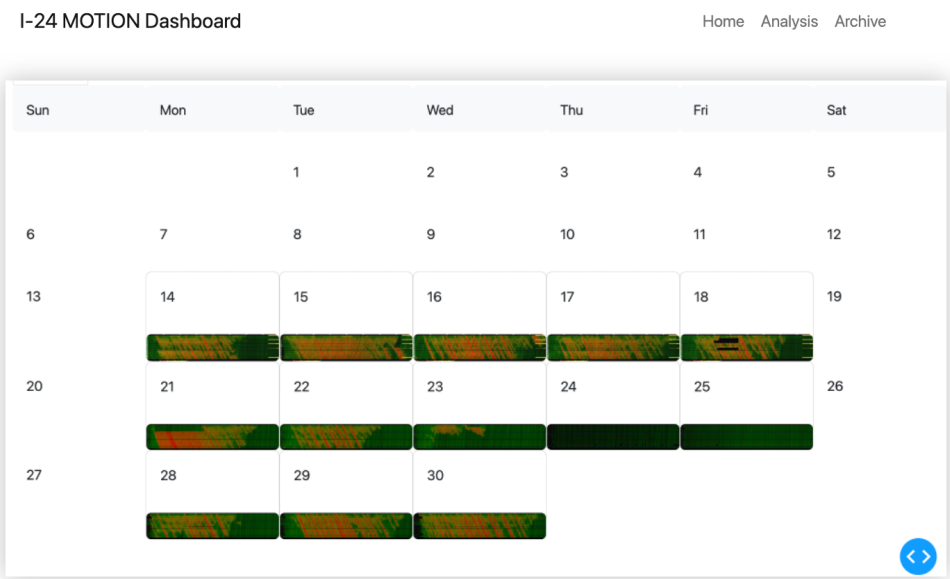


(d) Thursday Nov 24 2022 (Thanksgiving)

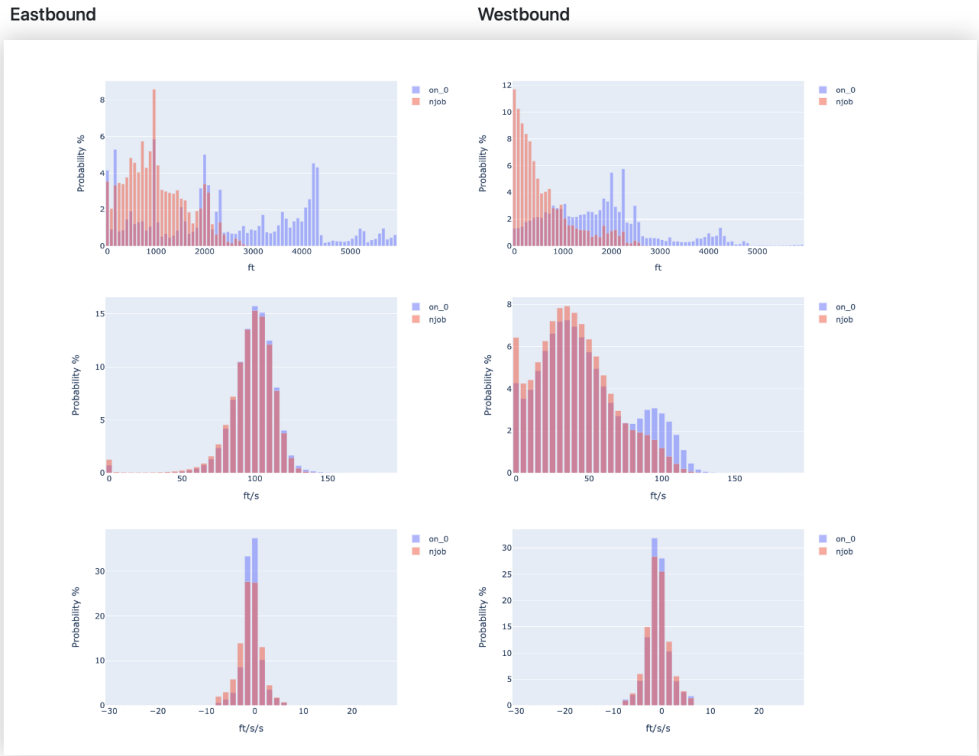


(e) Friday Nov 25 2022 (Black Friday)

Figure 7.14: Time-space diagrams of 5 days in November 2022. Photo credit: Gergely Zachar



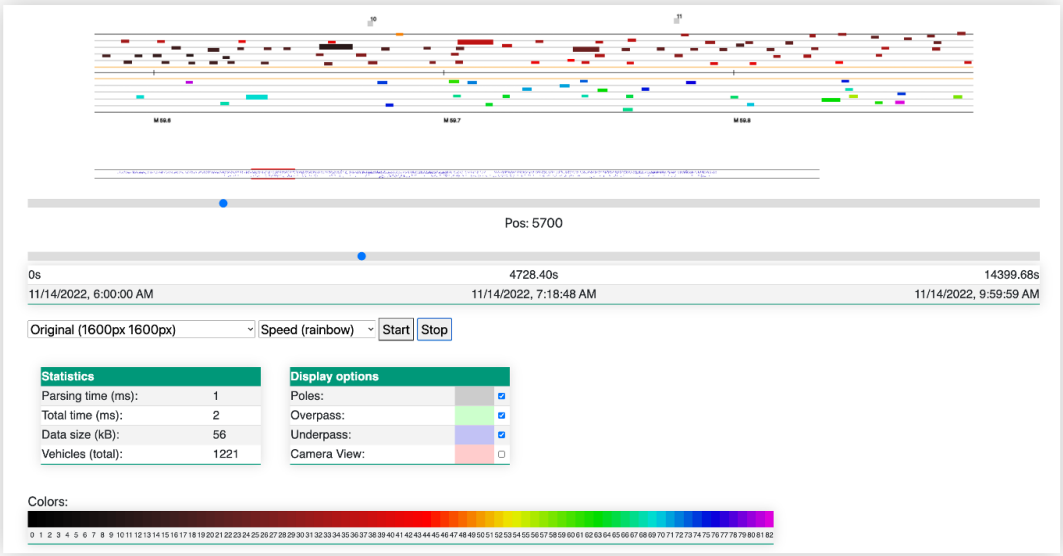
(a) A calendar view of November, 2022



(b) Traffic dynamics. Top: trajectory lengths, middle: speed, bottom: acceleration

Figure 7.15: Dashboard example (1/2)

(a) Statistics examples. Top: vehicle type distribution, bottom: trajectory OD matrix



(b) An overhead animation. Photo credit: Gergely Zachar

Figure 7.16: Dashboard example (2/2)

Interactive dashboards

During the data generation development phase, it is crucial to perform qualitative evaluations of the data in a meaningful manner. To facilitate the visualization of qualitative measures, we have constructed an alpha-version analysis component. This component computes traffic-relevant statistics, such as speed distribution, vehicle type distributions, and the start and ending locations of each tracked trajectory. The resulting statistics are then presented through an internal dashboard system.

The current utilization of the dashboard system primarily involves comparing statistics across multiple data processes. Figure 7.15-7.16 offer illustrative examples of this comparison. The dashboard's landing page showcases a calendar view (Figure 7.15a), with thumbnail time-space diagrams for each day on which data has been produced. Upon selecting a specific day, users are directed to the "analysis" page (Figure 7.15b, 7.16a and 7.16b), where comprehensive statistics for all runs occurring on that day are displayed. These statistics include trajectory length, speed and acceleration distributions, vehicle type distribution, an OD matrix of all trajectories, and an overhead animation view of any selected road segment and time range. These visualizations enable us to access fundamental traffic information, identify potential data processing issues, and facilitate further enhancements. In the future, additional information can be integrated into the dashboard system to analyze various aspects, such as crashes, ramp traffic, and corridor control strategies. We also intend to make the dashboards publicly accessible alongside future data releases.

7.5. Conclusion

High-quality trajectory data can close the gap for understanding microscopic traffic phenomena. A real-world live testbed like I-24 MOTION helps researchers to understand the impact of mixed autonomy in traffic. However, data produced by cameras and upstream computer vision algorithms still lacks high quality to be ready for research use. In this chapter we demonstrate a two-step data postprocessing pipeline to automatically reconcile detection and tracking data. The pipeline includes a fragment association algorithm to solve an online min-cost flow problem, and a trajectory rectification approach formulated as a quadratic programming. The accuracy is benchmarked on a set of numerical experiments using micro-simulation, as well as on two manually-labeled ground truth datasets. Results show that the two-step treatments improve a variety of trajectory quality measures on all the testing cases, given different traffic scenarios. Noticeably, it significantly improves the velocity and acceleration dynamics, despite the parameter tuning and cost function design step are only preliminary. This proposed pipeline has high promises to replace previous manual efforts on data cleaning.

For future work, a few algorithmic improvements can be made on existing methods to potentially address issues in real data. For example, different cost models for data association and an exhaustive parameter tuning need to be performed to improve the current results. The cost model design should consider more expressive features, such as appearance embedding (colors, dimensions and type of vehicles) in addition to the dynamics information, to fully describe the similarity between two fragments. Furthermore, a multi-vehicle reconciliation formulation can be considered to account for potential collisions.

CHAPTER 8

Conclusions and future work

8.1. A recap of contributions

In this thesis, we address the challenges of reconstructing complex traffic dynamics from both microscopic and macroscopic scales. We apply modeling, parameter and state estimation techniques to solve traffic dynamics reconstruction problems, and build a software pipeline for large-scale traffic data generation.

A recap of the contributions are summarized below:

Novel parameter identifiability analysis for car-following systems. Chapter 3 provides a comprehensive practical and theoretical analysis of parameter identifiability for car-following models expressed as ODEs. Two methods are developed: a numerical direct test to find worst-case indistinguishable parameters and a differential geometry-based approach for structural identifiability. The results show that car-following models are structurally identifiable in generic, noise-free settings, but practical identifiability challenges may arise under specific experimental conditions or with noisy data.

Efficient identification of ACC and human-driving car-following dynamics. A novel online filtering method for discovering ACC car-following systems (Chapter 4) and an end-to-end data-driven model for human drivers (Chapter 5) are developed. The proposed methods demonstrate improved computation efficiency and better dynamics recovery compared to existing techniques. The online system identification method based on recursive least squares (RLS) shows fast and accurate recovery of ACC car-following dynamics. The Gaussian Process (GP) model for human-driving outperforms existing ACCs in a human-in-the-loop driving simulation platform.

Accurate traffic state estimation in heterogeneous traffic. A novel multi-class traffic estimation formulation is introduced in Chapter 6 to estimate density evolution for each vehicle class. Innovative particle filter algorithms are designed to estimate complex mixed traffic evolution using sparse and noisy data, accommodating nonlinear and non-differentiable state evolution dynamics. Real traffic data from a mixed-traffic corridor validates the accurate recovery of traffic density evolution for each vehicle class.

Automatic trajectory data reconciliation pipeline for traffic testbed. A data reconciliation pipeline is proposed in Chapter 7 to improve the video-based trajectory extraction system accuracy. The pipeline automatically imputes, smooths, and corrects erroneous trajectory measurements using an online object matching algorithm and a convex optimization formulation. The resulting high-quality trajectory data from the MOTION testbed benefits mixed-autonomy traffic research and other transportation studies.

8.2. Limitations and potential improvements

This dissertation addresses the challenge of reconstructing traffic dynamics at both the microscopic level, focusing on modeling car-following dynamics of ACC and human drivers, and the macroscopic level, solving state estimation of traffic density evolution. However, certain limitations still remain unaddressed, which are important to acknowledge.

Accurately modeling human drivers, who constitute a significant portion of road users, poses a primary challenge due to their inherent heterogeneity and variations influenced by diverse factors. Chapter 5 explores modeling human car-following behavior using Gaussian Process (GP) models. However, the purely data-driven approach, devoid of physical constraints, may lead to infeasible predictions and sensitivity to training data quality. To address this, adaptive training of the GP model can consider variations in driving behaviors over time and speed ranges as data accumulates. Incorporating human interaction and physical constraints as direct feedback during training can guide the model towards feasible solutions. Promising techniques like physics-informed Gaussian Process [262, 163] or other time-series and dynamics modeling approaches (e.g., Long Short-term Memory (LSTM) [90] and inverse reinforcement learning [155]) offer potential enhancements. Additionally, leveraging contextual information such as weather and road geometry can expand the model's functionality, accommodating a broader range of driving scenarios.

Despite significant progress in addressing data quality issues from video-based tracking data, additional challenges remain. In Chapter 7, the proposed data association approach substantially reduces fragmentation and enhances trajectory lengths. However, the association results currently rely solely on motion information in the formulation of cost modeling (Equation 7.11). To improve this aspect, future work could explore more expressive features, such as appearance embeddings (colors, dimensions, and vehicle types), in addition to dynamics information to better describe the similarity between two fragments. Additionally, supervised learning approaches can be employed to learn the implicit matching cost from labeled datasets, further enhancing the data association process.

Lastly, there is room for improvement in the formulation for denoising and imputation of trajectory data, as presented in Chapter 7.3.2. The current approach solves a convex optimization problem for each individual trajectory, without considering interactions among multiple trajectories. Exploring alternative formulations that consider multiple trajectories, such as simultaneously enhancing individual trajectories while ensuring a safe distance between vehicles (conflict constraints), could be a promising avenue. Although this would make the problem non-convex, efficient solvers and heuristics can be explored to mitigate the computational load.

8.3. **Future research directions**

Below outlines potential future research directions built upon existing works.

Detecting safety-critical driving behaviors from large-scale traffic data: Investigating the relationship between individual drivers, whether human or automated vehicles, and its impact on traffic flow efficiency remains a crucial area of research. The use of large-scale traffic data, such as I-24 MOTION, can potentially shed light on this connection. There are three steps to advance this research. (1) Develop rigorous and comprehensive quantitative measures to assess the safety and efficiency of individual drivers. These measures should consider a range of factors, such as vehicle speed, lane-changing behavior, acceleration, and following distances, to capture the nuances of different driving styles and their impact on traffic dynamics. (2) Design novel algorithms to identify causal relationships between specific driving behaviors and their adverse effects on safety and traffic flow disruption. Causal inference techniques, such as causal Bayesian networks or counterfactual analysis, can be employed to establish cause-and-effect relationships from observational traffic data. (3) Explore the use of graph-based algorithms for outlier detection in traffic data. By modeling the

interactions of driving agents as a graph, outlier detection techniques can reveal unusual patterns or behaviors that deviate significantly from the norm, and signal potential adversarial driving actions.

Develop novel data-driven traffic models: Traditional traffic flow models based on partial differential equations (PDEs) often rely on simplified assumptions and fixed parameters, which may not capture the full complexity of real-world traffic dynamics. In addition, they typically treat traffic as a continuous quantity and may not account for individual driver characteristics. There are two possible remedies: first, data-driven models based on machine learning techniques, such as neural operators [126] and physics-informed neural networks [45, 31] that incorporate physical laws or constraints into neural networks, can learn complex patterns and non-linear relationships from traffic data and ensure the model's predictions align with fundamental traffic principles. It is important to investigate the possibility of implementing these data-driven models in real-time traffic management systems, which includes developing algorithms that can continuously learn from incoming data, adapting to changing traffic conditions and providing up-to-date predictions and insights. Second, agent-based models can be an avenue to increase granularity of traditional traffic flow models. They simulate individual vehicles or drivers as separate entities with their own behaviors and decision-making processes. This approach allows for the representation of heterogeneous traffic, human driver interactions, and individual variations in driving behavior.

Distributed traffic monitoring and control with connected vehicles: We are reaching a point where nearly every vehicle on the road is constantly collecting sensing data about the driver and the vehicle itself as well as the surrounding environment. It can be powerful to leverage the communication amongst all the vehicles to collaboratively optimize traffic flow, reduce congestion, and enhance safety. There are three tasks. First, design data fusion and aggregation techniques to combine the information collected from multiple vehicles to obtain a comprehensive view of traffic conditions. This can involve data assimilation from fixed location sensors as well as crowdsourced vehicle sensing data, to monitor infrastructure conditions and traffic conditions. This step promotes a wide coverage and timely updates, and increases data reliance. Second, we can investigate the use of consensus algorithms and federated learning [124] for distributed traffic control at intersections and merging lanes. In federated learning, models are trained locally on individual vehicles, and only aggregated model updates are shared, which ensures data privacy and reduces communication overhead. Challenges such as data imbalance amongst vehicles, computational resources allocation and robust diagnostics for outlier and faulty data, should be addressed in this step. The last task is simulation and real-world testing. We can simulate the distributed traffic monitoring and control system in various traffic scenarios using realistic traffic simulation platforms. Real-world testing and field trials need to be conducted to validate the effectiveness and scalability of the proposed algorithms and approaches.

APPENDIX A

Appendix for Chapter 3

Given the symbolic matrix (3.21), one can find numerical substitutions for all the symbolic variables such that (3.21) is full rank. Such substitutions belong to a generic initial condition and parameter set. One example is:

$$[k_1, k_2, \tau, u_0, v_0, s_0] = [0.01, 0.12, 1.4, 30, 33, 40]. \tag{A.1}$$

The corresponding identifiability matrix becomes:

$$\mathcal{O}_I(\tilde{x}_0, u) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -0.0100 & 0.134 & 6.20 & 3.00 & 0.330 \\ 0.00134 & -0.00796 & 1.579 & -0.824 & -0.0484 \\ -0.0000796 & -0.000274 & -0.658 & 0.107 & 0.003456 \end{bmatrix}, \tag{A.2}$$

and $\text{rank}(\mathcal{O}_I(\tilde{x}_0, u))$=5.

A.0.1. **Proof for proposition 3**

Recall the system of equations for CTH-RV:

$$\begin{aligned} \dot{x}(t) = \begin{bmatrix} \dot{s}(t) \\ \dot{v}(t) \end{bmatrix} &= \begin{bmatrix} u(t) - v(t) \\ k_1(s(t) - \tau v(t)) + k_2(u(t) - v(t)) \end{bmatrix} \\ y(t) &= x(t) \end{aligned} \tag{A.3}$$

This system in fact can be written as a scalar differential equation:

$$\dot{v}(t) = k_1 \int_0^t (u(\xi) - v(\xi))d\xi + (-k_1\tau - k_2)v(t) + k_2 u(t). \tag{A.4}$$

Take the derivative on both sides to get rid of the integral operator:

$$\ddot{v}(t) = k_1(u(t) - v(t))dt + (-k_1\tau - k_2)\dot{v}(t) + k_2\dot{u}(t). \tag{A.5}$$

Re-arrange to separate the input and output:

$$\ddot{v}(t) + (k_1\tau + k_2)\dot{v}(t) + k_1 v(t) = k_1 u(t) + k_2\dot{u}(t). \tag{A.6}$$

Now this is the standard form of a second-order non-homogeneous differential equation. The solution $v(t)$ has two parts, a homogeneous solution (or complementary solution $v_c(t)$, corresponds to the transient response), and a particular solution (or $V_p(t)$, corresponds to the steady state response). The general solution would be the sum of two parts:

$$v(t) = v_c(t) + V_p(t). \tag{A.7}$$

First let's see the homogeneous solution $v_c(t)$ by solving

$$\ddot{v}(t) + (k_1\tau + k_2)\dot{v}(t) + k_1 v(t) = 0. \tag{A.8}$$

The solution is of the form:

$$v_c(t) = c_1 e^{r_1 t} + c_2 e^{r_2 t}, \tag{A.9}$$

where r_1 and r_2 are the roots of the characteristic equation for (A.8), and c_1 and c_2 are constants, which can be solved using initial values after the particular solution $V_p(t)$ is obtained.

Moving on to $V_p(t)$. We will show the condition such that the steady state response $V_p(t)$ does not depend on k_1.

For simplicity, let us assume the input (forcing) function as a sinusoidal wave, i.e., $u(t) = a\sin(\omega t)$. Because we can represent almost all functions by Fourier series, which is a superposition of sine and cosine waves, the choice of $u(t)$ does not change the solution. The forcing term, or the RHS of (A.6) is

$$g(t) = k_1 u(t) + k_2 \dot{u}(t) = k_1 a\sin(\omega t) + k_2 a\omega\cos(\omega t). \tag{A.10}$$

By undetermined coefficients method, we can solve for the particular solution and the corresponding coefficients:

$$V_p(t) = A\sin(\omega t) + B\cos(\omega t), \tag{A.11}$$

where A, B can be solved by plugging $V_p(t)$ into (A.6). A, B can be represented in terms of a, ω and the model parameters θ. We will simplify the expressions by denoting A, B as functions of some parameters:

$$\begin{aligned} A &= f_A(a, \omega, \theta) \\ B &= f_B(a, \omega, \theta). \end{aligned} \tag{A.12}$$

Recall the superposition principle for ODE since ODE is a linear operator, the particular solution will always be a combination of sine and cosine waves with the same frequency but different magnitudes as the input $u(t)$. In this case, we just need to see if A, B depend on any parameter $p \in \{k_1, k_2, \tau\}$ to make a difference in the steady state response. This can be done by taking the partial derivatives $\dfrac{\partial f_A}{\partial p}$ and $\dfrac{\partial f_B}{\partial p}$ and examine the dependency. In other words:

$$\begin{aligned} &\text{find the solution } \mathcal{S}_p \text{ for} \\ &\frac{\partial f_A}{\partial p} = 0 \text{ and } \frac{\partial f_B}{\partial p} = 0, \ \forall\ p \in \{k_1, k_2, \tau\} \in \mathcal{P} \end{aligned} \tag{A.13}$$

where $\mathcal{P}$ is the set of legal values for parameter p. If the solution set for each p, $\mathcal{S}_p$, does not depend on p, then p is unidentifiable. Solve for (A.13) we get

$$\begin{aligned} \mathcal{S}_{k_1} &= \{k_2 = \frac{1}{\tau}\} \\ \mathcal{S}_{k_2} &= \emptyset \\ \mathcal{S}_{\tau} &= \emptyset. \end{aligned} \tag{A.14}$$

This suggests that A, B are independent of k_1 if $k_2 = \dfrac{1}{\tau}$.

Now let us go back to the initial condition and the transient response. In order to use the information $s(0) = s_0$, we'll need to work on the first-order differential equation (A.4) instead of (A.6). Simply plug in the initial condition $s(0) = s_0$ and $v(0) = v_0$ in (A.4) to obtain:

$$\dot{v}(0) = k_1 s_0 + (-k_1\tau - k_2)v_0 + k_2 u_0, \tag{A.15}$$

where $\int_0^t (u(\xi) - v(\xi))d\xi|_{t=0} = s_0$, and u_0 is the initial value of the input. Now if we do the same as before (A.13) by taking partial derivatives of $\dot{v}(0)$ to find the unidentifiable parameter(s) and the corresponding "condition" in order for that parameter to be unidentifiable, i.e.,

$$\begin{aligned} &\text{for } p \in \{k_1, k_2, \tau\}, \text{ find the solution } \mathcal{S}_p \text{ for} \\ &\frac{\partial \dot{v}(0)}{\partial p} = 0 \ \forall \ p \in \mathcal{P} \end{aligned} \tag{A.16}$$

Solving (A.16) we get:

$$\begin{aligned} \mathcal{S}_{k_1} &= \{s_0 = \tau v_0\} \\ \mathcal{S}_{k_2} &= \emptyset \\ \mathcal{S}_{\tau} &= \emptyset. \end{aligned} \tag{A.17}$$

The solution shows that k_1 cannot be identified if $\tau = \dfrac{s_0}{v_0}$ from the transient response. Substitute $\tau = \dfrac{s_0}{v_0}$ and $k_2 = 1/\tau = \dfrac{v_0}{s_0}$ into the $\mathcal{O}_I(\tilde{x}_0, u)$, we obtain:

$$\mathcal{O}_I(\tilde{x}_0, u) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ -k_1 & v_0/s_0 + k_1 s_0/v_0 & 0 & v_0 - u(t) & k_1 v_0 \\ -k_1(v_0/s_0 + k_1 s_0/v_0) & (v_0/s_0 + k_1 s_0/v_0)^2 - k_1 & 0 & o_{44} & o_{45} \\ o_{51} & o_{52} & 0 & o_{54} & o_{55} \end{bmatrix} \tag{A.18}$$

$$\begin{aligned} o_{44} &= -\dot{u}(t) - (v_0/s_0 + k_1 s_0/v_0)(v_0 - u(t)) - v_0(v_0 - u(t))/s_0 \\ o_{45} &= -k_1 v_0(v_0/s_0 + k_1 s_0/v_0) - k_1 v_0(v_0 - u(t))/s_0 \\ o_{51} &= k_1(k_1 - (v_0/s_0 + k_1 s_0/v_0)^2) \\ o_{52} &= -(v_0/s_0 + k_1 s_0/v_0)(k_1 - (v_0/s_0 + k_1 s_0/v_0)^2) - k_1(v_0/s_0 + k_1 s_0/v_0) \\ o_{54} &= \dot{u}(t)(2v_0/s_0 + k_1 s_0/v_0) - k_1(v_0 - u(t)) - (k_1 - (v_0/s_0 + k_1 s_0/v_0)^2) \\ &\quad (v_0 - u(t)) - \ddot{u}(t) + (v_0(2v_0/s_0 + 2k_1 s_0/v_0))/s_0 \\ o_{55} &= k_1 v_0 \dot{u}(t)/s_0 - k_1 v_0(k_1 - (v_0/s_0 + k_1 s_0/v_0)^2) - k_1^2(v_0 - u(t)) + \\ &\quad k_1 v_0(v_0/s_0 + k_1 s_0/v_0)(v_0 - u(t))^2/s_0. \end{aligned}$$

Matrix (A.18) clearly shows that the above substitution results in rank($\mathcal{O}_I(\tilde{x}_0, u)$)=4. The column corresponding to parameter k_1 is zero, meaning that k_1 is unidentifiable. This result is in agreement with the analytical proof that k_1 does not affect either the transient response or the steady state response. Note that we can choose $u_0 \neq v_0$ such that the initial condition of the system is at non-equilibrium. Please see Figure 3.2c for a visualization. Hence we complete the proof.

APPENDIX B

Appendix for Chapter 5

B.1. **Predictive Safety Filter**

A purely data-driven control approach such as GP does not explicitly take driving safety into account. Throughout the literature, we found learning-based control achieves "safe-by-design" with verified control envelopes [8], fixed-point computations of the set-valued mappings [160], and safety filtering [226], as common approaches. In this paper we adopt a predictive safety filtering approach similar to [226], which finds a safe acceleration profile that is closest to the GP-predicted acceleration and achieves collision avoidance.

Consider the following notations and assumptions:

Let $\mathcal{I}_{\geq k}$ denotes a set of integers in the interval $[k, \infty) \in \mathbb{R}$. Let a_k denote the acceleration of the leader vehicle at time k, which we assume can be measured. y_k stands for the acceleration for the follower at k. $a_{\min}$ denotes the hardest braking deceleration for the follower, which the follower vehicle can actuate instantaneously. $s_{\min}$ is the minimum space gap.

We develop a safety filter on commanded accelerations to achieve collision avoidance. The safety filter seeks to ensure two properties are met at all times. The first property is collision avoidance in the form $s_k \geq s_{\min}$. The second property is bounded deceleration in the form $y_k \geq a_{\min}$. To choose accelerations that achieve these two properties we form at each time-step k a set of allowable states into which the vehicle can move to time-step $k+1$, which we denote as $\mathcal{S}$:

$$(s_k, v_k, u_k, a_k) \in \mathcal{S} \Rightarrow y_{k'} \geq a_{\min}, s_{k'} \geq s_{\min}, \forall k^{'} \in \mathcal{I}_{\geq k} \tag{B.1}$$

according to the following discrete-time dynamics:

$$\begin{bmatrix} s \\ v \\ u \\ a \end{bmatrix}_{k+1} = g(s_k, v_k, u_k, a_k, y_k) = \begin{bmatrix} s \\ v \\ u \\ a \end{bmatrix}_k + \begin{bmatrix} u - v \\ y \\ a \\ 0 \end{bmatrix}_k \Delta t. \tag{B.2}$$

$\mathcal{S}$ is derived using a standard stopping time condition under constant acceleration from the leading vehicle (see appendix). By choosing y_k such that $(s_k, v_k, u_k, a_k) \in \mathcal{S}$ the filter ensures that either both collision avoidance and bounded acceleration will be met in all following time-steps, or $\mathcal{S} = \emptyset$ meaning that a collisions cannot be avoided. In the case that multiple such commanded accelerations exist, we choose the y_k that is closest to that prescribed by the GP $\hat{y}_k$. This can be stated in the following form:

$$\begin{aligned} \underset{y_k}{\text{minimize}} \ & (\hat{y}_k - y_k)^2 \\ \text{s.t. } & \hat{y}_k = f_{GP}(s_k, v_k, u_k, \theta) \\ & [s_{k+1}, v_{k+1}, u_{k+1}, a_{k+1}]^T = g(s_k, v_k, u_k, a_k, y_k) \\ & (s_{k+1}, v_{k+1}, u_{k+1}, a_{k+1}) \in \mathcal{S}. \end{aligned} \tag{B.3}$$

Scenarios in which the set $\mathcal{S} = \emptyset$ can be trivially triggered through simulating large lead vehicle decelerations that exceed $a_{\min}$ in magnitude. This is consistent with game theoretic results that

describe collisions between systems with equal, and unequal, dynamics and input ranges. Additional study of the feasibility of these safety regions through data and analysis of naturalistic and controlled scenarios is reserved for future work. One possible alternative formulation is through, for example, a *control barrier function* [5].

B.2. **Derivation of safe set**

For a car-following system shown in Figure 3.1, we derive the safe set $\mathcal{S}$ given the state (s_0, v_0, u_0, a_0) and $a_{\min}$, which stand for the initial space gap, follower velocity, leader velocity, leader acceleration and the hardest braking deceleration for the follower. The safe condition gives a requirement for (s_0, v_0, u_0, a_0) such that future collision can be prevented if the follower vehicle executes $a_{\min}$, $a_{\min} < 0$. Let the safety space-gap margin be $s_{\min}$, $s_{\min} > 0$, and assume the given state is safe, i.e., $s_0 \geq s_{\min}$, and $v_0 \geq 0, u_0 \geq 0$.

Consider two scenarios:

1. $a_0 < 0$ and
2. $a_0 \geq 0$.

Scenario 1): Denote the leader and follower position as $p_l(t)$ and $p_f(t)$, respectively. Consider a non-decreasing position for the leader vehicle when decelerating:

$$p_l(t) = \begin{cases} p_l(0) + u_0 t + \frac{1}{2} a_0 t^2, & 0 < t < -\frac{u_0}{a_0} \\ p_l(0) - \frac{u_0^2}{2a_0}, & t \geq -\frac{u_0}{a_0}, \end{cases}$$

Similarly, the non-decreasing follower's position during hardest braking can be denoted as

$$p_f(t) = \begin{cases} p_f(0) + v_0 t + \frac{1}{2} a_{\min} t^2, & 0 < t < -\frac{v_0}{a_{\min}} \\ p_f(0) - \frac{v_0^2}{2a_{\min}}, & t \geq -\frac{v_0}{a_{\min}}. \end{cases}$$

Denote the stopping time for leader and follower as

$$T_l^s = -\frac{u_0}{a_0}, T_f^s = -\frac{v_0}{a_{\min}}.$$

Safety requires that the space gap between the two vehicles is above the safety margin when both vehicles are at a stop, i.e.,

$$p_l(T_l^s) - p_f(T_f^s) - l > s_{\min}, \text{ or}$$

$$p_l(0) - \frac{u_0^2}{2a_0} - \left(p_f(0) - \frac{v_0^2}{2a_{\min}} \right) - l > s_{\min}$$

$$s_0 - \frac{u_0^2}{2a_0} + \frac{v_0^2}{2a_{\min}} > s_{\min}.$$

Consequently, the condition for safe state when $a_0 < 0$ is

$$C_1 := s_0 - \frac{u_0^2}{2a_0} + \frac{v_0^2}{2a_{\min}} > s_{\min}.$$

Scenario 2): The position for the leader becomes

$$p_l(t) = p_l(0) + u_0 t + \frac{1}{2} a_0 t^2.$$

The same safety criterion can be derived by setting

$$s(t) = p_l(t) - p_f(t) - l$$

$$= \frac{1}{2}(a_0 - a_{\min})t^2 + (u_0 - v_0)t + s_0 > s_{\min} \ \forall t > 0.$$

Note that $s(t)$ is a convex quadratic function. It can be observed that $s(t) > 0 \ \forall t$ if $s(t)$ has no real roots, i.e., $(u_0 - v_0)^2 - 2(a_0 - a_{\min})(s_0 - s_{\min}) < 0$, or the larger of the real roots < 0, i.e., $-(u_0 - v_0) - \sqrt{(u_0 - v_0)^2 - 2(a_0 - a_{\min})(s_0 - s_{\min})} < 0$. The corresponding safe condition becomes

$$C_2 := s_0 - (u_0 - v_0)^2 - 2(a_0 - a_{\min})(s_0 - s_{\min}) < 0 \cup$$

$$(u_0 - v_0) + \sqrt{(u_0 - v_0)^2 - 2(a_0 - a_{\min})(s_0 - s_{\min})} > 0$$

The overall set for the safe state is

$$\mathcal{S} = \{s_0 \geq s_{\min}, v_0, u_0 \geq 0 | ((a_0 < 0) \cap C_1) \cup ((a_0 \geq 0) \cap C_2)\}.$$

APPENDIX C

Appendix for Chapter 7

C.1. **Proof for correctness of online negative cycle canceling**

Recall the negative cycle optimality condition in Theorem 2, we need to prove the following lemma:

Lemma 2. The circulation in $G^+_{r,k}$ is optimal, i.e., there is no more negative cycles in $G^+_{r,k}$ for every k.

Proof. We prove by induction. The base case is $G^+_{r,0}$, which contains a single node s and therefore has no circulation nor negative cycle. During the first iteration, $G^-_{r,1}$ has one cycle: $s \to u_1 \to v_1 \to s$. If the cost of this cycle is positive, then $G^+_{r,1} = G^-_{r,1}$ and no more negative cycle remains in $G^+_{r,1}$. Otherwise if the cost for this cycle is negative, $G^+_{r,1}$ is $G^-_{r,1}$ with all edges reversed and costs negated, therefore the only cycle $G^+_{r,1}$ has a positive cost.

For the induction, we want to prove that given $G^+_{r,k-1}$ which has no negative cycle, $G^+_{r,k}$ remains optimal (no negative cycles) after pushing flow through the min-cost cycle Γ in $G^-_{r,k}$. Note that if Γ does not exist on $G^-_{r,k}$, i.e., $G^+_{r,k} = G^-_{r,k}$, then $G^+_{r,k}$ remains optimal. If Γ exists, it is obvious that Γ must contain the subpath $u_k \to v_k \to s$ (one of the three scenarios illustrated in Figure 7.5-7.7). We proceed the proof by contradiction.

Suppose there exists a negative-cost cycle Δ in $G^+_{r,k}$. Let $\bar{\Gamma}$ be the residual cycle in $G^+_{r,k}$ obtained by reversing and negating the cost of Γ. If $\bar{\Gamma}$ and Δ share no common edges, then Δ must not contain the subpath $u_k \to v_k \to s$, nor any incident edges to u_k. This is because any flow that goes through an incident edge to u_k must come out through the edge $u_k \to v_k$ per the flow conservation constraint. Therefore if Δ exists it must have already existed in $G^+_{r,k-1}$, which contradicts the precondition that $G^+_{r,k-1}$ is optimal (no negative cycles).

On the other hand if there exists a subpath $\pi(u, v) \in \Gamma$ and the residual path $\pi(v, u) \in \Delta$, i.e., $\bar{\Gamma}$ and Δ share a common subpath $\pi(v, u)$, we can prove that Γ is not the min-cost cycle in $G^-_{r,k}$. Let the subpath $\pi(u, v)$ and $\Gamma^{'}$ form the cycle Γ, and the subpath $\pi(v, u)$ and $\Delta^{'}$ form the cycle Δ (see Figure C.1). We have $\$(\pi(u, v)) = -\$(\pi(v, u))$ given the definition of a residual graph, along with the assumptions that Γ and Δ are negative cost:

$$\$(\Gamma) = \$(\pi(u, v)) + \$(\Gamma^{'}) < 0, \text{and}$$

$$\$(\Delta) = \$(\Delta^{'}) - \$(\pi(v, u)) < 0,$$

We can get

$$\$(\Gamma^{'}) + \$(\Delta^{'}) = \$(\Gamma) + \$(\Delta),$$

meaning that the cycle formed by $\Gamma^{'}$ and $\Delta^{'}$ in $G^-_{r,k}$ has a lower cost than either Γ or Δ. It contradicts the fact that Γ is the least-cost cycle on $G^-_{r,k}$.

Therefore we proved that $G^+_{r,k}$ obtained from each iteration in Algorithm 6 must be optimal. □

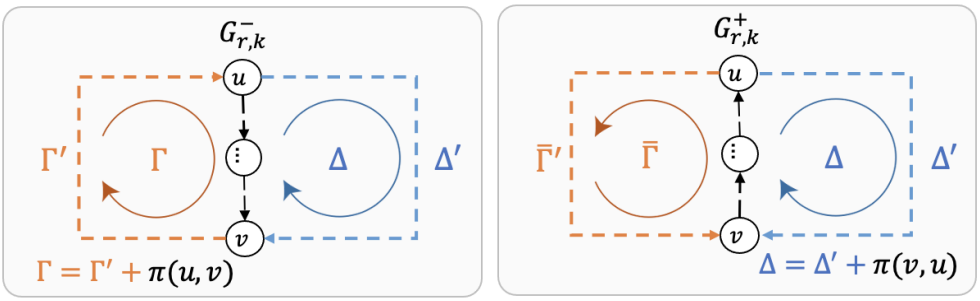


Figure C.1: Proof that the larger cycle composed of Δ' and Γ' is of lesser cost than Γ in $G^-_{r,k}$.

C.2. **Improvements for online NCC**

We outline a few improvements on the runtime and memory of running the online NCC algorithm in practice.

C.2.1. **Runtime improvements**

The offline NCC algorithm has time complexity of $\boldsymbol{O}(|V||E|^2 \log |V|)$, given that the step of finding the minimum mean cycle (the cycle whose average cost per edge is smallest) takes $\boldsymbol{O}(|V||E|)$. The FindMinCycle step in the online NCC algorithm can be further improved based on the following result from the proof in the previous section:

Corollary 2. If a negative cycle exists on $G^-_{r,k}$ after adding the new fragment ϕ_k, then it must contain a subpath $\pi(u_k, s) = (u_k \to v_k \to s)$.

This observation is helpful because we can limit our search for the min-cost cycle at each iteration to include this subpath. In order to find Γ, which has the cost of $\$(\Gamma) = \$(\pi(s, u_k)) + \$(\pi(u_k, s))$, we simply need to find the shortest path from s to u_k in $G^-_{r,k}$ and check if $\$(\pi(s, u_k)) + \$(\pi(u_k, s)) < 0$, as there is only one path for $\pi(u_k, s)$ and the cost of which is fixed. The step of FindMinCycle can be reduced to finding the single-source shortest path, which reduced the runtime to $\Theta(|E| + |V| \log |V|)$ at every iteration.

C.2.2. **Memory bound**

To limit the size of the graph at each iteration, we add a step CleanGraph($G^+_{r,k}, \tau$) to remove the trajectory (circulation) that is timed out at a customized time threshold τ. Since all the fragments are added in order of time, we can simply check the tails of each trajectory, or the succeeding nodes to the dummy node s at each residual graph $G^+_{r,k}$ for timeout. If timeout exceeds τ, all the nodes along the entire circulation (except for s) can be safely removed from $G^+_{r,k}$.

The removal of a circulation keeps the remaining flow in $G^+_{r,k}$ feasible because according to Lemma 1, removing one cycle does not interfere with other cycles as no cycles have shared edges. The remaining of $G^+_{r,k}$ is still optimal because no negative cycle can be created in a subgraph of an optimal residual graph.

Algorithm 7 Memory-bounded online NCC

Input: Set of fragments $\Phi = \{\phi_i\}$
Result: Set of trajectories $T = \{\tau_i\}$

$f \leftarrow 0$
$G^+_{r,0} \leftarrow \{s\}$
$k \leftarrow 1$
$\tau \leftarrow$ A time window
for *each* ϕ_k *(ordered by last timestamp)* **do**
  $G^-_{r,k} \leftarrow \text{AddNode}(G^+_{r,k-1}\ ,\ \phi_k)$
  $\Gamma \leftarrow \text{FindMinCycle}(G^-_{r,k})$
  $G^+_{r,k} \leftarrow \text{PushFlow}(G^-_{r,k}, \Gamma)$
  $G^+_{r,k} \leftarrow \text{CleanGraph}(G^+_{r,k}, \tau)$
  $k \leftarrow k + 1$
end
$T \leftarrow \text{FlowToTrajectories}(G^+_{r,k})$

BIBLIOGRAPHY

[1] F. Abodo, A. Berthaume, S. Zitzow-Childs, and L. Bobadilla. Strengthening the case for a bayesian approach to car-following model calibration and validation using probabilistic programming. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 4360–4367, 2019. doi: 10.1109/ITSC.2019.8917416.

[2] ACC. Adaptive cruise control datasets. https://acc-dataset.github.io/datasets/, 2019.

[3] R. K. Ahuja and J. B. Orlin. Combinatorial algorithms for inverse network flow problems. *Networks: An International Journal*, 40(4):181–187, 2002.

[4] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pages 1424–1431. IEEE, 2014.

[5] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.

[6] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431. IEEE, 2019.

[7] M. Andriluka, S. Roth, and B. Schiele. People-tracking-by-detection and people-detection-by-tracking. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. doi: 10.1109/CVPR.2008.4587583.

[8] N. Aréchiga and B. Krogh. Using verified control envelopes for safe controller design. In *2014 American Control Conference*, pages 2918–2923, 2014. doi: 10.1109/ACC.2014.6859307.

[9] C. Audet and J. E. Dennis. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13(3):889–903, 2002. doi: 10.1137/S1052623400378742. URL https://doi.org/10.1137/S1052623400378742.

[10] A. Aw and M. Rascle. Resurrection of "second order" models of traffic flow. *SIAM Journal on Applied Mathematics*, 60(3):916–938, 2000. doi: 10.1137/S0036139997332099. URL https://doi.org/10.1137/S0036139997332099.

[11] R. Babuška. Neuro-fuzzy methods for modeling and identification. In *Recent Advances in Intelligent Paradigms and Applications*, pages 161–186. Springer, 2003.

[12] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. Structure stability of congestion in traffic dynamics. *Japan Journal of Industrial and Applied Mathematics*, 11:203–223, 1994.

[13] M. Bando, K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Phys. Rev. E*, 51:1035–1042, Feb 1995. doi: 10.1103/PhysRevE.51.1035. URL https://link.aps.org/doi/10.1103/PhysRevE.51.1035.

[14] M. Bando, H. K., A. Nakayama, A. Shibata, and Y. Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Physical Review E*, 51(2):1035–1042, 1995.

[15] Z. Bareket, P. S. Fancher, Huei Peng, Kangwon Lee, and C. A. Assaf. Methodology for assessing adaptive cruise control behavior. *IEEE Transactions on Intelligent Transportation Systems*, 4(3):123–131, 2003.

[16] E. Barmpounakis and N. Geroliminis. On the new era of urban traffic monitoring with massive drone data: The pneuma large-scale field experiment. *Transportation research part C: emerging technologies*, 111:50–71, 2020.

[17] D. Beckmann, M. H. Riva, M. Dagen, and T. Ortmaier. Comparison of online-parameter estimation methods applied to a linear belt drive system. In *2016 European Control Conference (ECC)*, pages 364–369, June 2016. doi: 10.1109/ECC.2016.7810312.

[18] R. Bellman and K. Åström. On structural identifiability. *Mathematical Biosciences*, 7(3): 329–339, 1970. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(70)90132-X. URL http://www.sciencedirect.com/science/article/pii/002555647090132X.

[19] G. Bellu, M. Saccomani, S. Audoly, and L. D'Angiò. Daisy: A new software tool to test global identifiability of biological and physiological systems. *Computer Methods and Programs in Biomedicine*, 88:52–61, 11 2007. doi: 10.1016/j.cmpb.2007.07.002.

[20] S. Benzoni-Gavage and R. M. Colombo. An n-populations model for traffic flow. *European Journal of Applied Mathematics*, 14:587–612, 2002.

[21] O. Bernard, Z. Hadj-Sadok, D. Dochain, A. Genovesi, and J.-P. Steyer. Dynamical model development and parameter identification for an anaerobic wastewater treatment process. *Biotechnology and Bioengineering*, 75(4):424–438, 2001. doi: 10.1002/bit.10036. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/bit.10036.

[22] K. Bernardin and R. Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing*, 2008:1–10, 2008.

[23] B. Besselink and K. H. Johansson. String stability and a delay-based spacing policy for vehicle platoons subject to disturbances. *IEEE Transactions on Automatic Control*, 62(9):4376–4391, Sep. 2017. ISSN 0018-9286. doi: 10.1109/TAC.2017.2682421.

[24] S. Blandin, A. Couque, A. Bayen, and D. Work. On sequential data assimilation for scalar macroscopic traffic flow models. *Physica D: Nonlinear Phenomena*, 241(17):1421–1440, 2012. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2012.05.005. URL

http://www.sciencedirect.com/science/article/pii/S0167278912001339.

[25] R. Boel and L. Mihaylova. A compositional stochastic model for real time freeway traffic simulation. *Transportation Research Part B: Methodological*, 40(4):319–334, May 2006. URL https://ideas.repec.org/a/eee/transb/v40y2006i4p319-334.html.

[26] A. W. Bowman and A. Azzalini. *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations*, volume 15. New York: Oxford University Press Inc., 2000.

[27] S. Boyd and S. Sastry. Necessary and sufficient conditions for parameter convergence in adaptive control. *Automatica*, 22(6):629–639, 1986. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(86)90002-6. URL http://www.sciencedirect.com/science/article/pii/0005109886900026.

[28] S. Boyd, S. P. Boyd, and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[29] S. L. Brunton, J. L. Proctor, and J. N. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[30] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A multimodal dataset for autonomous driving. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11618–11628, 2020.

[31] S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

[32] G. Castnnón and L. Finn. Multi-target tracklet stitching through network flows. In *2011 Aerospace Conference*, pages 1–7. IEEE, 2011.

[33] C. Chalons, M. L. D. Monache, and P. Goatin. A conservative scheme for non-classical solutions to a strongly coupled PDE-ODE problem. *Interfaces and Free Boundaries*, 19(4):553–570, 2017. URL https://hal.inria.fr/hal-01070262.

[34] R. E. Chandler, R. Herman, and E. W. Montroll. Traffic dynamics: Studies in car following. *Operations Research*, 6(2):165–184, 1958. doi: 10.1287/opre.6.2.165. URL https://doi.org/10.1287/opre.6.2.165.

[35] M. J. Chappell and K. R. Godfrey. Structural identifiability of the parameters of a nonlinear batch reactor model. *Mathematical Biosciences*, 108(2):241–251, 1992. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(92)90058-5. URL http://www.sciencedirect.com/science/article/pii/0025556492900585.

[36] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

[37] X. Chen and L. Shi. *Stochastic evolutions of dynamic traffic flow: Modeling and applications*. Springer-Verlag Berlin Heidelberg, 2015. doi: 10.1007/978-3-662-44572-3.

[38] Z. Chen. Bayesian filtering: From kalman filters to particle filters, and beyond. *Statistics*, 182, Jan 2003. doi: 10.1080/02331880309257.

[39] F. A. Chiarello and P. Goatin. Non-local multi-class traffic flow models. *Networks and Heterogeneous Media*, 14:371, Apr 2019. doi: 10.3934/nhm.2019015.

[40] C.-Y. Chong. Graph approaches for data association. In *2012 15th international conference on information fusion*, pages 1578–1585. IEEE, 2012.

[41] B. Ciuffo and V. Punzo. "No Free Lunch" theorems applied to the calibration of traffic simulation models. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):553–562, 2014. doi: 10.1109/TITS.2013.2287720.

[42] B. Coifman and L. Li. A critical evaluation of the next generation simulation (ngsim) vehicle trajectory dataset. *Transportation Research Part B: Methodological*, 105:362–377, 2017.

[43] B. Coifman, M. Wu, K. Redmill, and D. A. Thornton. Collecting ambient vehicle trajectories from an instrumented probe vehicle: High quality data for microscopic traffic flow studies. *Transportation Research Part C: Emerging Technologies*, 72:254–271, 2016.

[44] G. Costeseque and A. Duret. Mesoscopic multiclass traffic flow modeling on multi-lane sections. In *Proceedings of the 95th annual meeting transportation research board - TRB*, page 27, Washington DC, United States, Jan. 2016. URL https://hal.archives-ouvertes.fr/hal-01250438.

[45] S. Cuomo, V. S. Di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022.

[46] C. F. Daganzo. The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B: Methodological*, 28(4):269–287, 1994. ISSN 0191-2615. doi: https://doi.org/10.1016/0191-2615(94)90002-7. URL http://www.sciencedirect.com/science/article/pii/0191261594900027.

[47] C. F. Daganzo. The cell transmission model, part II: Network traffic. *Transportation Research Part B: Methodological*, 29(2):79–93, 1995. ISSN 0191-2615. doi: https://doi.org/10.1016/0191-2615(94)00022-R. URL http://www.sciencedirect.com/science/article/pii/019126159400022R.

[48] F. De Souza and R. Stern. Calibrating microscopic car-following models for adaptive cruise control vehicles: Multiobjective approach. *Journal of Transportation Engineering, Part A: Systems*, 147(1):04020150, 2021.

[49] M. L. Delle Monache, T. Liard, A. Rat, R. Stern, R. Bhadani, B. Seibold, J. Sprinkle, D. B. Work, and B. Piccoli. *Feedback Control Algorithms for the Dissipation of Traffic Waves with Autonomous Vehicles*. Springer International Publishing, Cham, 2019. ISBN 978-3-030-25446-9. doi: 10.1007/978-3-030-25446-9_12. URL https://doi.org/10.1007/978-3-030-25446-9_12.

[50] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[51] A. Doucet and A. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of Nonlinear Filtering*, 12, Jan 2009.

[52] A. Duret, C. Buisson, and N. Chiabaut. Estimating individual speed-spacing relationship and assessing ability of Newell's car-following model to reproduce trajectories. *Transportation Research Record*, 2088(1):188–197, 2008. doi: 10.3141/2088-20. URL https://doi.org/10.3141/2088-20.

[53] A. Duret, C. Buisson, and N. Chiabaut. Estimating individual speed-spacing relationship and assessing ability of newell's car-following model to reproduce trajectories. *Transportation research record*, 2088(1):188–197, 2008.

[54] H. Eren, S. Makinist, E. Akin, and A. Yilmaz. Estimating driving behavior by a smartphone. In *2012 IEEE Intelligent Vehicles Symposium*, pages 234–239. IEEE, 2012.

[55] H. Fan and M. S. Poole. What is personalization? perspectives on the design and implementation of personalization in information systems. *Journal of Organizational Computing and Electronic Commerce*, 16(3-4):179–202, 2006.

[56] S. Fan and D. B. Work. A heterogeneous multiclass traffic flow model with creeping. *SIAM Journal on Applied Mathematics*, 75(2):813–835, 2015. ISSN 0036-1399. doi: 10.1137/140977977.

[57] A. Ferrara, S. Simona, and S. Silvia. *Freeway traffic modelling and control*. Springer International Publishing, 2018. ISBN 9783319759593. doi: 10.1007/978-3-319-75961-6.

[58] G. C. Ferro and V. Gerdt. Improved kolchin–ritt algorithm. *Programming and Computer Software*, 29:83–87, 2004.

[59] C. Fiori, K. Ahn, and H. A. Rakha. Power-based electric vehicle energy consumption model: Model development and validation. *Applied Energy*, 168:257–268, 2016.

[60] C. Fiori, V. Marzano, V. Punzo, and M. Montanino. Energy consumption modeling in presence

of uncertainty. *IEEE Transactions on Intelligent Transportation Systems*, 22(10):6330–6341, 2020.

[61] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian journal of Mathematics*, 8:399–404, 1956.

[62] M. Garavello, K. Han, and B. Piccoli. *Models for Vehicular Traffic on Networks*, volume 9 of *AIMS Series on Applied Mathematics*. American Institute of Mathematical Sciences, 2016.

[63] M. Garavello, K. Han, and B. Piccoli. *Models for Vehicular Traffic on Networks*. American Institute of Mathematical Sciences, 2016. ISBN 978-1-60133-019-2.

[64] N. H. Gartner, C. J. Messer, and A. Rathi. Traffic flow theory-a state-of-the-art report: revised monograph on traffic flow theory. *Transportation Research International Documentation*, 2002.

[65] S. Gashaw, P. Goatin, and J. Härri. Modeling and Analysis of Mixed Flow of Cars and Powered Two Wheelers. *Transportation research. Part C, Emerging technologies*, pages 1–44, 2018. URL https://hal.inria.fr/hal-01708005.

[66] D. C. Gazis and C. H. Knapp. On-line estimation of traffic densities from time-series of flow and speed data. *Transportation Science*, 5(3):283–301, 1971. doi: 10.1287/trsc.5.3.283. URL https://doi.org/10.1287/trsc.5.3.283.

[67] D. C. Gazis, R. Herman, and R. B. Potts. Car-following theory of steady-state traffic flow. *Operations Research*, 7(4):499–505, 1959. doi: 10.1287/opre.7.4.499. URL https://doi.org/10.1287/opre.7.4.499.

[68] D. C. Gazis, R. Herman, and R. B. Potts. Car-following theory of steady-state traffic flow. *Operations Research*, 7(4):499–505, 1959.

[69] D. C. Gazis, R. Herman, and R. W. Rothery. Nonlinear follow-the-leader models of traffic flow. *Operations Research*, 9(4):545–567, 1961. ISSN 0030364X, 15265463. URL http://www.jstor.org/stable/167126.

[70] P. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, 1981. ISSN 0191-2615. doi: https://doi.org/10.1016/0191-2615(81)90037-0. URL https://www.sciencedirect.com/science/article/pii/0191261581900370.

[71] P. G. Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, 1981.

[72] A. Girard. *Approximate methods for propagation of uncertainty with Gaussian process models*. PhD thesis, Citeseer, 2004.

[73] S. T. Glad. Solvability of differential algebraic equations and inequalities: An algorithm. In *1997 European Control Conference (ECC)*, pages 1195–1200, 1997. doi: 10.23919/ECC.1997.7082262.

[74] D. Gloudemans and D. B. Work. Vehicle tracking with crop-based detection. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 312–319. IEEE, 2021.

[75] D. Gloudemans, Y. Wang, J. Ji, G. Zachar, W. Barbour, D. B. Work, E. Hall, M. Cebelak, and L. Smith. I-24 motion: An instrument for freeway traffic science. *Transportation Research Part C: Emerging Technologies*, 2023. Under review.

[76] S. Godunov. A-difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Mat. Sb. (N.S.)*, 47(89):271–306, 1959. URL http://mi.mathnet.ru/msb4873.

[77] A. V. Goldberg and R. E. Tarjan. Finding minimum-cost circulations by canceling negative cycles. *Journal of the ACM (JACM)*, 36(4):873–886, 1989.

[78] M. Grewal and K. Glover. Identifiability of linear and nonlinear dynamical systems. *IEEE Transactions on Automatic Control*, 21(6):833–837, 1976.

[79] G. Gunter, D. Gloudemans, R. E. Stern, S. McQuade, R. Bhadani, M. Bunting, M. L. Delle Monache, R. Lysecky, B. Seibold, J. Sprinkle, B. Piccoli, and D. B. Work. Are commercially implemented adaptive cruise control systems string stable? *arXiv:1905.02108*, 2019.

[80] G. Gunter, C. Janssen, W. Barbour, R. Stern, and D. Work. Model based string stability of adaptive cruise control systems using field data. *IEEE Transactions on Intelligent Vehicles*, pages 1–1, 2019. ISSN 2379-8858. doi: 10.1109/TIV.2019.2955368.

[81] G. Gunter, R. Stern, and D. B. Work. Modeling adaptive cruise control vehicles from experimental data: model comparison. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3049–3054, 2019.

[82] G. Gunter, D. Gloudemans, R. E. Stern, S. McQuade, R. Bhadani, M. Bunting, M. L. Delle Monache, R. Lysecky, B. Seibold, J. Sprinkle, B. Piccoli, and D. B. Work. Are commercially implemented adaptive cruise control systems string stable? *IEEE Transactions on Intelligent Transportation Systems*, 22(11):6992–7003, 2021. doi: 10.1109/TITS.2020.3000682.

[83] Q. Guo, L. Li, and X. J. Ban. Urban traffic signal control with connected and automated vehicles: A survey. *Transportation research part C: emerging technologies*, 101:313–334, 2019.

[84] M. Hasenjäger, M. Heckmann, and H. Wersing. A survey of personalization for advanced driver assistance systems. *IEEE Transactions on Intelligent Vehicles*, 5(2):335–344, 2019.

[85] Z. He, L. Zheng, and W. Guan. A simple nonparametric car-following model driven by field data. *Transportation Research Part B: Methodological*, 80:185–201, 2015. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2015.07.010. URL http://www.sciencedirect.com/science/article/pii/S0191261515001575.

[86] A. Hegyi, D. Girimonte, R. Babuska, and B. D. Schutter. A comparison of filter configurations for freeway traffic state estimation. In *Proceedings of 2006 IEEE Intelligent Transportation Systems Conference*, pages 1029–1034, Sep 2006. doi: 10.1109/ITSC.2006.1707357.

[87] D. Helbing, M. Treiber, A. Kesting, and M. Schönhof. Theoretical vs. empirical classification and prediction of congested traffic states. *European Physics Journal B*, 3 2009. doi: 10.1140/epjb/e2009-00140-5. URL http://arxiv.org/abs/0903.0929http://dx.doi.org/10.1140/epjb/e2009-00140-5.

[88] R. Hermann and A. Krener. Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, 22(5):728–740, 1977.

[89] P. D. Hill. Kernel estimation of a distribution function. *Communications in Statistics - Theory and Methods*, 14(3):605–620, 1985. doi: 10.1080/03610928508828937. URL https://doi.org/10.1080/03610928508828937.

[90] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

[91] Y. Hollander and R. Liu. The principles of calibrating traffic microsimulation models. *Transportation*, 35(3):347–362, May 2008. doi: 10.1007/s11116-007-9156-2. URL https://ideas.repec.org/a/kap/transp/v35y2008i3p347-362.html.

[92] H. Hong, A. Ovchinnikov, G. Pogudin, and C. Yap. Global identifiability of differential models. *Communications on Pure and Applied Mathematics*, 73(9):1831–1879, 2020. doi: https://doi.org/10.1002/cpa.21921. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.21921.

[93] S. Hoogendoorn and R. Hoogendoorn. Calibration of microscopic traffic-flow models using multiple data sources. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 368:4497–517, 10 2010. doi: 10.1098/rsta.2010.0189.

[94] S. P. Hoogendoorn and R. Hoogendoorn. Generic calibration framework for joint estimation of car-following models by using microscopic data. *Transportation Research Record*, 2188(1): 37–45, 2010. doi: 10.3141/2188-05. URL https://doi.org/10.3141/2188-05.

[95] Y.-X. Huang, R. Jiang, H. Zhang, M.-B. Hu, J.-F. Tian, B. Jia, and Z.-Y. Gao. Experimental study and modeling of car-following behavior under high speed situation. *Transportation research part C: emerging technologies*, 97:194–215, 2018.

[96] P. Ioannou, Z. Xu, S. Eckert, D. Clemons, and T. Sieja. Intelligent cruise control: theory

and experiment. In *Proceedings of 32nd IEEE Conference on Decision and Control*, pages 1885–1890 vol.2, Dec 1993. doi: 10.1109/CDC.1993.325521.

[97] R. Jiang, M.-B. Hu, H. M. Zhang, Z.-Y. Gao, B. Jia, Q.-S. Wu, B. Wang, and M. Yang. Traffic experiment reveals the nature of car-following. *PloS one*, 9(4):e94351, 2014.

[98] R. Jiang, M.-B. Hu, H. Zhang, Z.-Y. Gao, B. Jia, and Q.-S. Wu. On some experimental features of car-following behavior and how to model them. *Transportation Research Part B: Methodological*, 80:338–354, 2015. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2015.08.003. URL https://www.sciencedirect.com/science/article/pii/S0191261515001782.

[99] X. Jin, Y. Zhang, F. Wang, L. Li, D. Yao, Y. Su, and Z. Wei. Departure headways at signalized intersections: A log-normal distribution model approach. *Transportation Research Part C: Emerging Technologies*, 17(3):318–327, 2009. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2009.01.003. URL http://www.sciencedirect.com/science/article/pii/S0968090X09000102.

[100] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(7):1409–1422, 2012. doi: 10.1109/TPAMI.2011.239.

[101] V. Kanagaraj, G. Asaithambi, T. Toledo, and T.-C. Lee. Trajectory data and flow characteristics of mixed traffic. *Transportation Research Record*, 2491(1):1–11, 2015. doi: 10.3141/2491-01. URL https://doi.org/10.3141/2491-01.

[102] Y. Kang, H. Yin, and C. Berger. Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments. *IEEE Transactions on Intelligent Vehicles*, 4(2):171–185, 2019. doi: 10.1109/TIV.2018.2886678.

[103] J. Karlsson, M. Anguelova, and M. Jirstrand. An efficient method for structural identifiability analysis of large dynamic systems. *IFAC Proceedings Volumes*, 45(16):941–946, 2012. ISSN 1474-6670. doi: https://doi.org/10.3182/20120711-3-BE-2027.00381. URL http://www.sciencedirect.com/science/article/pii/S1474667015380745. 16th IFAC Symposium on System Identification.

[104] B. S. Kerner and H. Lieu. The physics of traffic: Empirical freeway pattern features, engineering applications; and theory. *Physics Today*, 58(11):54–56, 2005.

[105] A. Kesting and M. Treiber. Calibrating car-following models by using trajectory data: Methodological study. *Transportation Research Record*, 2088(1):148–156, 2008. doi: 10.3141/2088-16. URL https://doi.org/10.3141/2088-16.

[106] A. Kesting and M. Treiber. Calibrating car-following models by using trajectory data: Methodological study. *Transportation Research Record*, 2088(1):148–156, 2008. doi: 10.3141/2088-16. URL https://doi.org/10.3141/2088-16.

[107] A. Kesting, M. Treiber, and D. Helbing. Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 368(1928):4585–4605, 2010.

[108] P. K. Khosla and T. Kanade. Parameter identification of robot dynamics. In *1985 24th IEEE Conference on Decision and Control*, pages 1754–1760, 1985.

[109] M. J. Klein. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14:205–220, 1966.

[110] V. L. Knoop, M. Wang, I. Wilmink, D. M. Hoedemaeker, M. Maaskant, and E.-J. Van der Meer. Platoon of SAE level-2 automated vehicles on public roads: Setup, traffic interactions, and stability. *Transportation Research Record*, page 0361198119845885, 2019.

[111] J. Kocijan and D. Petelin. Output-error model training for gaussian process models. In A. Dobnikar, U. Lotrič, and B. Šter, editors, *Adaptive and Natural Computing Algorithms*, pages 312–321, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-20267-4.

[112] J. Kocijan, A. Girard, B. Banko, and R. Murray-Smith. Dynamic systems identification with gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems*, 11(4):411–424, 2005. doi: 10.1080/13873950500068567. URL https://doi.org/10.1080/13873950500068567.

[113] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125, 2018. doi: 10.1109/ITSC.2018.8569552.

[114] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE, 2018.

[115] C. Kreutz. An easy and efficient approach for testing identifiability. *Bioinformatics (Oxford, England)*, 34(11):1913—1921, June 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty035. URL https://doi.org/10.1093/bioinformatics/bty035.

[116] M. Kuderer, S. Gulati, and W. Burgard. Learning driving styles for autonomous vehicles from demonstration. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2641–2646, 2015. doi: 10.1109/ICRA.2015.7139555.

[117] V. Kurtc and M. Treiber. Calibrating the local and platoon dynamics of car-following models on the reconstructed ngsim data. In V. L. Knoop and W. Daamen, editors, *Traffic and Granular Flow '15*, pages 515–522. Springer International Publishing, 2016. ISBN 978-3-319-33482-0.

[118] J. A. Laval, C. S. Toth, and Y. Zhou. A parsimonious model for the formation of oscillations in car-following models. *Transportation Research Part B: Methodological*, 70:228–238, 2014. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2014.09.004. URL http://www.sciencedirect.com/science/article/pii/S0191261514001581.

[119] P. Lenz, A. Geiger, and R. Urtasun. Followme: Efficient online min-cost flow tracking with bounded memory and computation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4364–4372, 2015.

[120] L. Li and X. M. Chen. Vehicle headway modeling and its inferences in macroscopic/microscopic traffic flow theory: A survey. *Transportation Research Part C: Emerging Technologies*, 76: 170–188, 2017. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2017.01.007. URL http://www.sciencedirect.com/science/article/pii/S0968090X17300141.

[121] L. Li, X. M. Chen, and L. Zhang. A global optimization algorithm for trajectory data based car-following model calibration. *Transportation Research Part C: Emerging Technologies*, 68:311–332, 2016. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2016.04.011. URL http://www.sciencedirect.com/science/article/pii/S0968090X16300249.

[122] L. Li, R. Jiang, Z. He, X. M. Chen, and X. Zhou. Trajectory data-based traffic flow studies: A revisit. *Transportation Research Part C: Emerging Technologies*, 114:225–240, 2020. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2020.02.016. URL http://www.sciencedirect.com/science/article/pii/S0968090X19312987.

[123] L. Li, R. Jiang, Z. He, X. M. Chen, and X. Zhou. Trajectory data-based traffic flow studies: A revisit. *Transportation Research Part C: Emerging Technologies*, 114:225–240, 2020. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2020.02.016. URL https://www.sciencedirect.com/science/article/pii/S0968090X19312987.

[124] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020.

[125] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybridboosted multi-target tracker for crowded scene. In *2009 IEEE conference on computer vision and pattern recognition*, pages 2953–2960. IEEE, 2009.

[126] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

[127] C.-Y. Liang and H. Peng. Optimal adaptive cruise control with guaranteed string stability. *Vehicle System Dynamics*, 32(4-5):313–330, 1999. doi: 10.1076/vesd.32.4.313.2083. URL https://www.tandfonline.com/doi/abs/10.1076/vesd.32.4.313.2083.

[128] M. J. Lighthill and G. B. Whitham. On kinematic waves ii. a theory of traffic flow on long

crowded roads. In *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, volume 229, pages 317–345, 1955. doi: 10.1098/rspa.1955.0089. URL https://royalsocietypublishing.org/doi/abs/10.1098/rspa.1955.0089.

[129] Y. Liu, Z. Wang, K. Han, Z. Shou, P. Tiwari, and J. H. L. Hansen. Sensor fusion of camera and cloud digital twin information for intelligent vehicles. In *IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2020.

[130] L. Ljung. *System Identification: Theory for the User*. Prentice Hall information and system sciences series. Prentice Hall PTR, 1999. ISBN 9780136566953. URL https://books.google.com/books?id=nHFoQgAACAAJ.

[131] L. Ljung and T. Glad. On global identifiability for arbitrary model parametrizations. *Automatica*, 30(2):265–276, 1994. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(94)90029-9. URL http://www.sciencedirect.com/science/article/pii/0005109894900299.

[132] J. Ma, C. Schwarz, Z. Wang, M. Elli, G. Ros, and Y. Feng. New simulation tools for training and testing automated vehicles. In G. Meyer and S. Beiker, editors, *Road Vehicle Automation 7*, pages 111–119, Cham, 2020. Springer International Publishing. ISBN 978-3-030-52840-9.

[133] T. Ma and B. Abdulhai. Genetic algorithm-based optimization approach and generic tool for calibrating traffic microscopic simulation parameters. *Transportation Research Record*, 1800 (1):6–15, 2002. doi: 10.3141/1800-02. URL https://doi.org/10.3141/1800-02.

[134] T. Maiwald, H. Hass, B. Steiert, J. Vanlier, R. Engesser, A. Raue, F. Kipkeew, H. H. Bock, D. Kaschek, C. Kreutz, and J. Timmer. Driving the model to its limit: Profile likelihood based model reduction. *PLOS ONE*, 11(9):1–18, 09 2016. doi: 10.1371/journal.pone.0162366. URL https://doi.org/10.1371/journal.pone.0162366.

[135] M. Makridis, G. Fontaras, B. Ciuffo, and K. Mattas. Mfc free-flow model: Introducing vehicle dynamics in microsimulation. *Transportation Research Record*, 2673(4):762–777, 2019. doi: 10.1177/0361198119838515. URL https://doi.org/10.1177/0361198119838515.

[136] M. Makridis, K. Mattas, A. Anesiadou, and B. Ciuffo. openacc. an open database of car-following experiments to study the properties of commercial acc systems, 2020.

[137] M. Makridis, K. Mattas, B. Ciuffo, F. Re, A. Kriston, F. Minarini, and G. Rognelund. Empirical study on the properties of adaptive cruise control systems and their impact on traffic flow and string stability. *Transportation Research Record*, 2674(4):471–484, 2020. doi: 10.1177/0361198120911047. URL https://doi.org/10.1177/0361198120911047.

[138] C. Mallikarjuna and K. R. Rao. Cellular automata model for heterogeneous traffic. *Journal of Advanced Transportation*, 43(3):321–345, 2009. doi: 10.1002/atr.5670430305. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/atr.5670430305.

[139] MATLAB Global Optimization Toolbox. Matlab global optimization toolbox, R2020a. The MathWorks, Natick, MA, USA.

[140] N. Meshkat, C. E.-z. Kuo, and J. DiStefano, III. On finding and using identifiable parameter combinations in nonlinear dynamic systems biology models and combos: A novel web implementation. *PLOS ONE*, 9(10):1–14, 10 2014. doi: 10.1371/journal.pone.0110261. URL https://doi.org/10.1371/journal.pone.0110261.

[141] N. Metropolis and S. Ulam. The monte carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949. ISSN 01621459. URL http://www.jstor.org/stable/2280232.

[142] H. Miao, X. Xia, A. S. Perelson, and H. Wu. On identifiability of nonlinear ode models and applications in viral dynamics. *SIAM Review*, 53(1):3–39, 2011. doi: 10.1137/090757009. URL https://doi.org/10.1137/090757009.

[143] L. Mihaylova, R. K. Boel, and A. Hegyi. Freeway traffic estimation within particle filtering framework. *Automatica*, 43:290–300, Feb 2007. doi: 10.1016/j.automatica.2006.08.023.

[144] L. Mihaylova, A. Hegyi, A. Gning, and R. K. Boel. Parallelized particle and gaussian sum particle filters for large-scale freeway traffic systems. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):36–48, Mar 2012. ISSN 1524-9050. doi: 10.1109/TITS.2011.2178833.

[145] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[146] V. Milanés and S. E. Shladover. Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data. *Transportation Research Part C: Emerging Technologies*, 48:285–300, 2014.

[147] V. Milanés, S. E. Shladover, J. Spring, C. Nowakowski, H. Kawazoe, and M. Nakamura. Cooperative adaptive cruise control in real traffic situations. *IEEE Transactions on Intelligent Transportation Systems*, 15(1):296–305, 2014.

[148] R. Mohan and G.Ramadurai. Multi-class Traffic Flow Model Based on Three-Dimensional Flow-Concentration Surface. In *Proceedings of the 98th annual meeting transportation research board - TRB*, pages 19–04534, Washington DC, United States, Jan. 2019. URL https://trid.trb.org/view/1573120.

[149] M. Montanino and V. Punzo. Trajectory data reconstruction and simulation-based validation against macroscopic traffic patterns. *Transportation Research Part B: Methodological*, 80: 82–106, 2015.

[150] J. Monteil and M. Bouroche. Robust parameter estimation of car-following models considering practical non-identifiability. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 581–588, Nov 2016. doi: 10.1109/ITSC.2016.7795612.

[151] J. Monteil, R. Billot, J. Sau, C. Buisson, and N.-E. El Faouzi. Calibration, estimation and sampling issues of car following model. In *Proceedings of the 92nd annual meeting of the Transportation Research Board*, 2014.

[152] J. Monteil, N. O'Hara, V. Cahill, and M. Bouroche. Real-time estimation of drivers' behaviour. *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 2046–2052, 2015.

[153] J. Monteil, M. Bouroche, and D. J. Leith. $\mathcal{L}_2$ and $\mathcal{L}_\infty$ stability analysis of heterogeneous traffic with application to parameter optimization for the control of automated vehicles. *IEEE Transactions on Control Systems Technology*, 27(3):934–949, 2018.

[154] R. Nair, H. S. Mahmassani, and E. Miller-Hooks. A porous flow approach to modeling heterogeneous traffic in disordered systems. *Procedia - Social and Behavioral Sciences*, 17: 611–627, 2011. ISSN 1877-0428. doi: https://doi.org/10.1016/j.sbspro.2011.04.534. URL http://www.sciencedirect.com/science/article/pii/S1877042811010937.

[155] A. Y. Ng, S. Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[156] D. Ngoduy. Applicable filtering framework for online multiclass freeway network estimation. *Physica A: Statistical Mechanics and its Applications*, 387(2-3):599–616, 2008.

[157] D. Ngoduy and R. Liu. Multiclass first-order simulation model to explain non-linear traffic phenomena. *Physica A: Statistical Mechanics and its Applications*, 385(2):667–682, 2007. ISSN 0378-4371. doi: https://doi.org/10.1016/j.physa.2007.07.041. URL http://www.sciencedirect.com/science/article/pii/S0378437107007911.

[158] D. Ngoduy and A. Sumalee. Adaptive estimation of noise covariance matrices in unscented Kalman filter for multiclass traffic flow model. *Transportation Research Record*, 2188(1):119–130, 2010. doi: 10.3141/2188-13. URL https://doi.org/10.3141/2188-13.

[159] NGSIM. The next generation simulation program. http://ops.fhwa.dot.gov/trafficanalysistools/ngsim.htm, 2006. U.S. Department of Transportation Federal Highway Administation.

[160] P. Nilsson, O. Hussien, A. Balkan, Y. Chen, A. D. Ames, J. W. Grizzle, N. Ozay, H. Peng, and P. Tabuada. Correct-by-construction adaptive cruise control: Two approaches. *IEEE Transactions on Control Systems Technology*, 24(4):1294–1307, 2016. doi: 10.1109/TCST.2015.2501351.

[161] G. Oh, D. J. Leblanc, and H. Peng. Vehicle energy dataset (ved), a large-scale dataset for vehicle energy consumption research. *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[162] L. Olivier, B. Huang, and I. Craig. Dual particle filters for state and parameter estimation with application to a run-of-mine ore mill. *Journal of Process Control*, 22(4):710–717, 2012. ISSN 0959-1524. doi: https://doi.org/10.1016/j.jprocont.2012.02.009. URL http://www.sciencedirect.com/science/article/pii/S0959152412000583.

[163] G. Pang and G. E. Karniadakis. Physics-informed learning machines for partial differential equations: Gaussian processes versus neural networks. *Emerging Frontiers in Nonlinear Science*, pages 323–343, 2020.

[164] S. Panwai and H. Dia. Comparative evaluation of microscopic car-following behavior. *IEEE Transactions on Intelligent Transportation Systems*, 6(3):314–325, 2005. doi: 10.1109/TITS.2005.853705.

[165] I. Papamichail, N. Bekiaris-Liberis, A. I. Delis, D. Manolis, K.-S. Mountakis, I. K. Nikolos, C. Roncoli, and M. Papageorgiou. Motorway traffic flow modelling, estimation and control with vehicle automation and communication systems. *Annual Reviews in Control*, 48:325–346, 2019. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2019.09.002. URL http://www.sciencedirect.com/science/article/pii/S1367578819300963.

[166] V. Papathanasopoulou and C. Antoniou. Towards data-driven car-following models. *Transportation Research Part C: Emerging Technologies*, 55:496–509, 2015. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2015.02.016. URL http://www.sciencedirect.com/science/article/pii/S0968090X15000716. Engineering and Applied Sciences Optimization (OPT-i) - Professor Matthew G. Karlaftis Memorial Issue.

[167] H. Pohjanpalo. System identifiability based on the power series expansion of the solution. *Mathematical Biosciences*, 41(1):21–33, 1978. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(78)90063-9. URL http://www.sciencedirect.com/science/article/pii/0025556478900639.

[168] N. Polson and V. Sokolov. Bayesian particle tracking of traffic flows. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):345–356, Feb 2018. doi: 10.1109/TITS.2017.2650947.

[169] V. Punzo and F. Simonelli. Analysis and comparison of microscopic traffic flow models with real traffic microscopic data. *Transportation Research Record*, 1934(1):53–63, 2005. doi: 10.1177/0361198105193400106. URL https://doi.org/10.1177/0361198105193400106.

[170] V. Punzo, M. T. Borzacchiello, and B. Ciuffo. On the assessment of vehicle trajectory data accuracy and application to the next generation simulation (ngsim) program data. *Transportation Research Part C: Emerging Technologies*, 19(6):1243–1262, 2011. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2010.12.007. URL https://www.sciencedirect.com/science/article/pii/S0968090X10001701.

[171] V. Punzo, B. Ciuffo, and M. Montanino. Can results of car-following model calibration based

on trajectory data be trusted? *Transportation Research Record*, 2315:11–24, 2012.

[172] V. Punzo, M. Montanino, and B. Ciuffo. Do we really need to calibrate all the parameters? variance-based sensitivity analysis to simplify microscopic traffic flow models. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):184–193, Feb 2015. ISSN 1524-9050. doi: 10.1109/TITS.2014.2331453.

[173] V. Punzo, M. Montanino, and B. Ciuffo. Do we really need to calibrate all the parameters? variance-based sensitivity analysis to simplify microscopic traffic flow models. *IEEE Transactions on Intelligent Transportation Systems*, 16(1):184–193, Feb 2015. ISSN 1524-9050. doi: 10.1109/TITS.2014.2331453.

[174] R. Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[175] L. Rakai, H. Song, S. Sun, W. Zhang, and Y. Yang. Data association in multiple object tracking: A survey of recent techniques. *Expert Systems with Applications*, 192:116300, 4 2022. ISSN 09574174. doi: 10.1016/j.eswa.2021.116300. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417421016031.

[176] C. E. Rasmussen. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-28650-9. doi: 10.1007/978-3-540-28650-9_4. URL https://doi.org/10.1007/978-3-540-28650-9_4.

[177] A. Raue, C. Kreutz, T. Maiwald, J. Bachmann, M. Schilling, U. Klingmüller, and J. Timmer. Structural and practical identifiability analysis of partially observed dynamical models by exploiting the profile likelihood. *Bioinformatics*, 25(15):1923–1929, 06 2009. ISSN 1367-4803. doi: 10.1093/bioinformatics/btp358. URL https://doi.org/10.1093/bioinformatics/btp358.

[178] P. I. Richards. Shock waves on the highway. *Operations Research*, 4(1):42–51, 1956. doi: 10.1287/opre.4.1.42. URL https://doi.org/10.1287/opre.4.1.42.

[179] M. A. Risso, N. Bhouri, A. J. Rubiales, and P. A. Lotito. A constrained filtering algorithm for freeway traffic state estimation. *Transportmetrica A: Transport Science*, 16(2):316–336, 2020. doi: 10.1080/23249935.2018.1549618. URL https://doi.org/10.1080/23249935.2018.1549618.

[180] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *European conference on computer vision*, pages 17–35. Springer, 2016.

[181] J. F. Ritt. Differential algebra. *American Mathematical Society Colloquium Publications*, 33, 1950. doi: http://dx.doi.org/10.1090/coll/033.

[182] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, et al. LGSVL simulator: A high fidelity simulator for autonomous driving. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems*

(ITSC), pages 1–6. IEEE, 2020.

[183] M. P. Saccomani, S. Audoly, and L. D'Angiò. Parameter identifiability of nonlinear systems: the role of initial conditions. *Automatica*, 39(4):619–632, 2003.

[184] W. J. Schakel, V. L. Knoop, and B. van Arem. Integrated lane change model with relaxation and synchronization. *Transportation Research Record*, 2316(1):47–57, 2012.

[185] T. Seo, A. M. Bayen, T. Kusakabe, and Y. Asakura. Traffic state estimation on highway: A comprehensive survey. *Annual Reviews in Control*, 43:128–151, 2017. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2017.03.005. URL http://www.sciencedirect.com/science/article/pii/S1367578817300226.

[186] X. Shi, D. Zhao, H. Yao, X. Li, D. K. Hale, and A. Ghiasi. Video-based trajectory extraction with deep learning for high-granularity highway simulation (high-sim). *Communications in Transportation Research*, 1:100014, 12 2021. ISSN 27724247. doi: 10.1016/j.commtr.2021.100014. URL https://linkinghub.elsevier.com/retrieve/pii/S2772424721000147.

[187] S. E. Shladover, C. A. Desoer, J. K. Hedrick, M. Tomizuka, J. Walrand, W.-B. Zhang, D. H. McMahon, H. Peng, S. Sheikholeslam, and N. McKeown. Automated vehicle control developments in the path program. *IEEE Transactions on Vehicular Technology*, 40(1):114–130, 1991.

[188] D. Simon. *Optimal State Estimation*. John Wiley & Sons, Inc., 2006. ISBN 9780471708582. doi: 10.1002/0470045345.

[189] P. T. Sokkalingam, R. K. Ahuja, and J. B. Orlin. New polynomial-time cycle-canceling algorithms for minimum-cost flows. *Networks*, 36(1):53–63, 2000. doi: https://doi.org/10.1002/1097-0037(200008)36:1<53::AID-NET6>3.0.CO;2-Y.

[190] P. Spannaus, P. Zechel, and K. Lenz. Automatum data: Drone-based highway dataset for the development and validation of automated driving software for research and commercial applications. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 1372–1377. IEEE, 2021.

[191] M. Stepančič and J. Kocijan. Gaussian process model-based system identification toolbox for matlab, 2017. URL https://github.com/Dynamic-Systems-and-GP/GPdyn.

[192] R. Stern, G. Gunter, and D. B. Work. Modeling and assessing adaptive cruise control stability: experimental insights. In *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 1–8, 2019.

[193] R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, H. Pohlmann, F. Wu, B. Piccoli, et al. Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments. *Transportation Research Part C: Emerging Tech-*

nologies, 89:205–221, 2018.

[194] R. E. Stern, S. Cui, M. L. D. Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, R. Haulcy, H. Pohlmann, F. Wu, B. Piccoli, B. Seibold, J. Sprinkle, and D. B. Work. Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments. *Transportation Research Part C: Emerging Technologies*, 89:205–221, 2018. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2018.02.005. URL http://www.sciencedirect.com/science/article/pii/S0968090X18301517.

[195] J. D. Stigter and J. Molenaar. A fast algorithm to assess local structural identifiability. *Automatica*, 58:118–124, 2015. ISSN 0005-1098. doi: https://doi.org/10.1016/j.automatica.2015.05.004. URL http://www.sciencedirect.com/science/article/pii/S0005109815001946.

[196] Y. Sugiyama, M. Fukui, M. Kikuchi, K. Hasebe, A. Nakayama, K. Nishinari, S. Tadaki, and S. Yukawa. Traffic jams without bottlenecks - experimental evidence for the physical mechanism of the formation of a jam. *New Journal of Physics*, 10(3):033001, mar 2008. doi: 10.1088/1367-2630/10/3/033001. URL https://doi.org/10.1088%2F1367-2630%2F10%2F3%2F033001.

[197] Y. Sun and D. Work. Scaling the kalman filter for large-scale traffic estimation. *IEEE Transactions on Control of Network Systems*, 5(3):968–980, 2017.

[198] Y. Sun and D. B. Work. Error bounds for Kalman filters on traffic networks. *Networks & Heterogeneous Media*, 13(2):261–295, 2018. doi: 10.3934/nhm.2018012.

[199] S. Surace, A. Kutschireiter, and J. Pfister. How to avoid the curse of dimensionality: Scalability of particle filters with and without importance weights. *SIAM Review*, 61(1):79–91, 2019. doi: 10.1137/17M1125340. URL https://doi.org/10.1137/17M1125340.

[200] D. Swaroop and J. Hedrick. String stability of interconnected systems. *IEEE Transactions on Automatic Control*, 41(3):349–357, 1996.

[201] M. W. Szeto and D. C. Gazis. Application of Kalman filtering to the surveillance and control of traffic systems. *Transportation Science*, 6(4):419–439, 1972. doi: 10.1287/trsc.6.4.419. URL https://doi.org/10.1287/trsc.6.4.419.

[202] A. Talebpour and H. S. Mahmassani. Influence of connected and autonomous vehicles on traffic flow stability and throughput. *Transportation Research Part C: Emerging Technologies*, 71:143–163, 2016.

[203] T. Tang, H. Huang, S. Zhao, and H. Shang. A new dynamic model for heterogeneous traffic flow. *Physics Letters A*, 373(29):2461–2466, 2009. ISSN 0375-9601. doi: https://doi.org/10.1016/j.physleta.2009.05.006. URL http://www.sciencedirect.com/science/article/pii/S0375960109005799.

[204] C. Thiemann, M. Treiber, and A. Kesting. Estimating acceleration and lane-changing dynamics from next generation simulation trajectory data. *Transportation Research Record*, 2088(1): 90–101, 2008.

[205] M. Treiber and A. Kesting. Microscopic calibration and validation of car-following models – a systematic approach. *Procedia - Social and Behavioral Sciences*, 80:922–939, 2013. ISSN 1877-0428. doi: https://doi.org/10.1016/j.sbspro.2013.05.050. URL http://www.sciencedirect.com/science/article/pii/S1877042813010197. 20th International Symposium on Transportation and Traffic Theory (ISTTT 2013).

[206] M. Treiber and A. Kesting. *Calibration and Validation*, pages 303–338. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-32460-4. doi: 10.1007/978-3-642-32460-4_16. URL https://doi.org/10.1007/978-3-642-32460-4_16.

[207] M. Treiber and A. Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 2013.

[208] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Phys. Rev. E*, 62(2):1805–1824, Aug 2000. doi: 10.1103/PhysRevE.62.1805. URL https://link.aps.org/doi/10.1103/PhysRevE.62.1805.

[209] M. Treiber, A. Kesting, and D. Helbing. Three-phase traffic theory and two-phase models with a fundamental diagram in the light of empirical stylized facts. *Transportation Research Part B: Methodological*, 44(8-9):983–1000, 2010.

[210] N. Tuncer and T. T. Le. Structural and practical identifiability analysis of outbreak models. *Mathematical Biosciences*, 299:1–18, 2018. ISSN 0025-5564. doi: https://doi.org/10.1016/j.mbs.2018.02.004. URL http://www.sciencedirect.com/science/article/pii/S0025556417303164.

[211] S. Vajda, K. R. Godfrey, and H. Rabitz. Similarity transformation approach to identifiability analysis of nonlinear compartmental models. *Mathematical Biosciences*, 93(2):217–248, 1989. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(89)90024-2. URL http://www.sciencedirect.com/science/article/pii/0025556489900242.

[212] C. P. van Hinsbergen, H. W. van Lint, S. P. Hoogendoorn, and H. J. van Zuylen. Bayesian calibration of car-following models. *IFAC Proceedings Volumes*, 42(15):91–97, 2009. ISSN 1474-6670. doi: https://doi.org/10.3182/20090902-3-US-2007.0049. URL http://www.sciencedirect.com/science/article/pii/S1474667016317803. 12th IFAC Symposium on Control in Transportation Systems.

[213] J. van Lint, S. P. Hoogendoorn, and A. Hegyi. Dual EKF state and parameter estimation in multi-class first-order traffic flow models. *IFAC Proceedings Volumes*, 41(2):14078–14083, 2008. ISSN 1474-6670. doi: https://doi.org/10.3182/20080706-5-KR-1001.02383. URL http://www.sciencedirect.com/science/article/pii/S1474667016412486.

[214] J. van Lint, S. P. Hoogendoorn, and M. Schreuder. Fastlane: New multiclass first-order traffic flow model. *Transportation Research Record: Journal of the Transportation Research Board*, 2088, Dec 2008. doi: 10.3141/2088-19.

[215] F. van Wageningen-Kessels, H. van Lint, K. Vuik, and S. Hoogendoorn. Genealogy of traffic flow models. *EURO Journal on Transportation and Logistics*, 4(4):445–473, Dec 2015. ISSN 2192-4384. doi: 10.1007/s13676-014-0045-5. URL https://doi.org/10.1007/s13676-014-0045-5.

[216] L. Vandenberghe. The cvxopt linear and quadratic cone program solvers. *Online: http://cvxopt. org/documentation/coneprog. pdf*, 2010.

[217] A. F. Villaverde. Observability and structural identifiability of nonlinear biological systems. *Complexity*, 2019:1–12, 2019. doi: 10.1155/2019/8497093.

[218] A. F. Villaverde and J. R. Banga. Structural properties of dynamic systems biology models: Identifiability, reachability, and initial conditions. *Processes*, 5(4):29, Jun 2017. ISSN 2227-9717. doi: 10.3390/pr5020029. URL http://dx.doi.org/10.3390/pr5020029.

[219] A. F. Villaverde and A. Barreiro. Identifiability of large nonlinear biochemical networks. *Match*, 76(2), 2016. ISSN 0340-6253. URL http://repositorium.sdum.uminho.pt/handle/1822/43872.

[220] A. F. Villaverde, A. Barreiro, and A. Papachristodoulou. Structural identifiability of dynamic systems biology models. *PLOS Computational Biology*, 12(10):1–22, 10 2016. doi: 10.1371/journal.pcbi.1005153. URL https://doi.org/10.1371/journal.pcbi.1005153.

[221] A. F. Villaverde, N. D. Evans, M. J. Chappell, and J. R. Banga. Input-dependent structural identifiability of nonlinear systems. *IEEE Control Systems Letters*, 3(2):272–277, 2019. doi: 10.1109/LCSYS.2018.2868608.

[222] A. F. Villaverde, N. Tsiantis, and J. R. Banga. Full observability and estimation of unknown inputs, states and parameters of nonlinear biological models. *Journal of The Royal Society Interface*, 16(156):20190043, 2019. doi: 10.1098/rsif.2019.0043. URL https://royalsocietypublishing.org/doi/abs/10.1098/rsif.2019.0043.

[223] C. Vivas, S. Siri, A. Ferrara, S. Sacone, G. Cavanna, and F. R. Rubio. Distributed consensus-based switched observers for freeway traffic density estimation. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 3445–3450. IEEE, 2015.

[224] N. Vyahhi, S. Bakiras, P. Kalnis, and G. Ghinita. Tracking moving objects in anonymized trajectories. In *International Conference on Database and Expert Systems Applications*, pages 158–171. Springer, 2008.

[225] K. P. Wabersich and M. N. Zeilinger. Linear model predictive safety certification for learning-

based control. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 7130–7135. IEEE, 2018.

[226] K. P. Wabersich and M. N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems, 2021.

[227] E. Walter and Y. Lecourtier. Unidentifiable compartmental models: what to do? *Mathematical Biosciences*, 56(1):1–25, 1981. ISSN 0025-5564. doi: https://doi.org/10.1016/0025-5564(81)90025-0. URL http://www.sciencedirect.com/science/article/pii/0025556481900250.

[228] E. Walter and Y. Lecourtier. Global approaches to identifiability testing for linear and nonlinear state space models. *Mathematics and Computers in Simulation*, 24(6):472–482, 1982. ISSN 0378-4754. doi: https://doi.org/10.1016/0378-4754(82)90645-0. URL https://www.sciencedirect.com/science/article/pii/0378475482906450.

[229] E. Walter, J. Norton, and L. Pronzato. *Identification of Parametric Models: From Experimental Data*. Communications and Control Engineering. Springer, 1997. ISBN 9782225853814. URL https://books.google.it/books?id=x_1QAAAAMAAJ.

[230] E. Walter, I. Braems, L. Jaulin, and M. Kieffer. Guaranteed numerical computation as an alternative to computer algebra for testing models for identifiability. In R. Alt, A. Frommer, R. B. Kearfott, and W. Luther, editors, *Numerical Software with Result Verification*, pages 124–131, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg. ISBN 978-3-540-24738-8.

[231] C. Wang, Y. Wang, Y. Wang, C.-T. Wu, and G. Yu. mussp: Efficient min-cost flow algorithm for multi-object tracking. *Advances in Neural Information Processing Systems*, 32, 2019.

[232] C. Wang, Y. Wang, and G. Yu. Efficient global multi-object tracking under minimum-cost circulation framework. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[233] H. Wang, W. Wang, J. Chen, and M. Jing. Using trajectory data to analyze intradriver heterogeneity in car-following. *Transportation Research Record*, 2188(1):85–95, 2010. doi: 10.3141/2188-10. URL https://doi.org/10.3141/2188-10.

[234] J. Wang, M. Lu, and K. Li. Characterization of longitudinal driving behavior by measurable parameters. *Transportation Research Record*, 2185(1):15–23, 2010.

[235] J. Wang, Y. Zheng, Q. Xu, J. Wang, and K. Li. Controllability analysis and optimal controller synthesis of mixed traffic systems. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1041–1047, 2019.

[236] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2007.

[237] M. Wang, W. Daamen, S. P. Hoogendoorn, and B. van Arem. Cooperative car-following control: Distributed algorithm and impact on moving jam features. *IEEE Transactions on Intelligent Transportation Systems*, 17(5):1459–1471, 2015.

[238] R. Wang, Y. Li, and D. B. Work. Comparing traffic state estimators for mixed human and automated traffic flows. *Transportation Research Part C: Emerging Technologies*, 78: 95–110, 2017. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2017.02.011. URL http://www.sciencedirect.com/science/article/pii/S0968090X17300517.

[239] X. Wang, R. Jiang, L. Li, Y. Lin, X. Zheng, and F.-Y. Wang. Capturing car-following behaviors by deep learning. *IEEE Transactions on Intelligent Transportation Systems*, 19(3):910–920, 2018. doi: 10.1109/TITS.2017.2706963.

[240] Y. Wang and M. Papageorgiou. Real-time freeway traffic state estimation based on extended kalman filter: a general approach. *Transportation Research Part B: Methodological*, 39(2): 141–167, 2005. ISSN 0191-2615. doi: https://doi.org/10.1016/j.trb.2004.03.003. URL http://www.sciencedirect.com/science/article/pii/S0191261504000438.

[241] Y. Wang and D. Work. Estimation for heterogeneous traffic using enhanced particle filters. *Transportmetrica A: Transport Science*, pages 1–26, 2021. URL https://doi.org/10.1080/23249935.2021.1881186.

[242] Y. Wang, G. Gunter, M. Nice, M. Delle Monache, and D. Work. Online parameter estimation methods for adaptive cruise control systems. *IEEE Transactions on Intelligent Vehicles*, 6(2): 288–298, 2020. URL https://doi.org/10.1109/TIV.2020.3023674.

[243] Y. Wang, G. Gunter, and D. B. Work. Online parameter estimation of adaptive cruise control models with delays and lags. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.

[244] Y. Wang, Z. Wang, K. Han, P. Tiwari, and D. Work. Personalized adaptive cruise control via gaussian process regression. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1496–1502, Indianapolis, IN, USA., 9 2021. IEEE. URL https://doi.org/10.1109/ITSC48978.2021.9564498.

[245] Y. Wang, M. L. Delle Monache, and D. B. Work. Identifiability of car-following dynamics. *Physica D: Nonlinear Phenomena*, 430:133090, 2022.

[246] Y. Wang, D. Gloudemans, Z. N. Teoh, L. Liu, G. Zachár, W. Barbour, and D. Work. Automatic vehicle trajectory data reconstruction at scale. *arXiv preprint arXiv:2212.07907*, 2022. doi: 10.48550/ARXIV.2212.07907. URL https://arxiv.org/abs/2212.07907.

[247] Y. Wang, Z. Wang, K. Han, P. Tiwari, and D. B. Work. Gaussian process-based personalized adaptive cruise control. *IEEE Transactions on Intelligent Transportation Systems*, 23(11): 21178–21189, 2022.

[248] Y. Wang, J. Ji, W. Barbour, and D. Work. Online min cost circulation for multi-object-tracking on fragments. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, 2023.

[249] Z. Wang, Y. Bian, S. E. Shladover, G. Wu, S. E. Li, and M. J. Barth. A survey on cooperative longitudinal motion control of multiple connected and automated vehicles. *IEEE Intelligent Transportation Systems Magazine*, 12(1):4–24, 2019.

[250] Z. Wang, G. Wu, K. Boriboonsomsin, M. Barth, et al. Cooperative ramp merging system: Agent-based modeling and simulation using game engine. *SAE International Journal of Connected and Automated Vehicles*, 2(2), 2019.

[251] Z. Wang, X. Liao, C. Wang, D. Oswald, G. Wu, K. Boriboonsomsin, M. Barth, K. Han, B. Kim, and P. Tiwari. Driver behavior modeling using game engine and real vehicle: A learning-based approach. *IEEE Transactions on Intelligent Vehicles*, 5(4):738–749, 2020.

[252] Z. Wang, K. Han, and P. Tiwari. Digital twin simulation of connected and automated vehicles with the unity game engine. In *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*, pages 1–4. IEEE, 2021.

[253] R. Wilson and J. Ward. Car-following models: fifty years of linear stability analysis – a mathematical perspective. *Transportation Planning and Technology*, 34(1):3–18, 2011. doi: 10.1080/03081060.2011.530826. URL https://doi.org/10.1080/03081060.2011.530826.

[254] D. B. Work, O.-P. Tossavainen, S. Blandin, A. M. Bayen, T. Iwuchukwu, and K. Tracton. An ensemble Kalman filtering approach to highway traffic estimation using GPS enabled mobile devices. In *Proceedings of 2008 47th IEEE Conference on Decision and Control*, pages 5062–5068, Dec 2008. doi: 10.1109/CDC.2008.4739016.

[255] M. Wright and R. Horowitz. Fusing loop and GPS probe measurements to estimate freeway density. *IEEE Transactions on Intelligent Transportation Systems*, 17(12):3577–3590, Dec 2016. ISSN 1524-9050. doi: 10.1109/TITS.2016.2565438.

[256] C. Wu, A. M. Bayen, and A. Mehta. Stabilizing traffic with autonomous vehicles. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6012–6018, 2018.

[257] X. Xia and C. H. Moog. Identifiability of nonlinear systems with application to hiv/aids models. *IEEE Transactions on Automatic Control*, 48(2):330–336, 2003.

[258] W. Xiang, H.-D. Tran, and T. T. Johnson. Output reachable set estimation and verification for multilayer neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 29(11):5777–5783, 2018.

[259] L. Xiao and F. Gao. Practical string stability of platoon of adaptive cruise control vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 12(4):1184–1194, 2011.

[260] L. Xiao, S. Darbha, and F. Gao. Stability of string of adaptive cruise control vehicles with parasitic delays and lags. In *2008 11th International IEEE Conference on Intelligent Transportation Systems*, pages 1101–1106, 2008.

[261] D. Yang, A. Kuijpers, G. Dane, and T. van der Sande. Impacts of large-scale truck platooning on dutch highways. *Transportation research procedia*, 37:425–432, 2019.

[262] X. Yang, G. Tartakovsky, and A. Tartakovsky. Physics-informed kriging: A physics-informed gaussian process regression method for data-model convergence. *arXiv preprint arXiv:1809.03461*, 2018.

[263] H. M. Zhang. A non-equilibrium traffic model devoid of gas-like behavior. *Transportation Research Part B: Methodological*, 36(3):275–290, Mar 2002. URL https://ideas.repec.org/a/eee/transb/v36y2002i3p275-290.html.

[264] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[265] P. Zhang, R.-X. LIU, S. C. WONG, and S.-Q. DAI. Hyperbolicity and kinematic waves of a class of multi-population partial differential equations. *European Journal of Applied Mathematics*, 17(2):171–200, 2006. doi: 10.1017/S095679250500642X.

[266] P. Zhang, S. Wong, and C.-W. Shu. A weighted essentially non-oscillatory numerical scheme for a multi-class traffic flow model on an inhomogeneous highway. *Journal of Computational Physics*, 212(2):739–756, 2006. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2005.07.019. URL http://www.sciencedirect.com/science/article/pii/S0021999105003542.

[267] P. Zhang, S. Wong, and Z. Xu. A hybrid scheme for solving a multi-class traffic flow model with complex wave breaking. *Computer Methods in Applied Mechanics and Engineering*, 197(45):3816–3827, 2008. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2008.03.003. URL http://www.sciencedirect.com/science/article/pii/S0045782508001175.

[268] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

[269] M. Čičić and K. H. Johansson. Stop-and-go wave dissipation using accumulated controlled moving bottlenecks in multi-class ctm framework. In *2019 IEEE 58th Conference on Decision and Control (CDC)*, pages 3146–3151, 2019.